学习网站开发　成就高薪梦想

典藏版



网站开发指南

Study the Guide with us

HTML+CSS
网页设计指南

书中所涉及的案例均为作者原创

详细介绍了CSS+DIV制作页面的主流方法

由简到难，由原始到整个体系来叙述图书知识结构

实例丰富，书中包含4个案例和120个小实例

从不一样的角度来分析HTML的应用规则

注重实际应用，理论与案例相结合

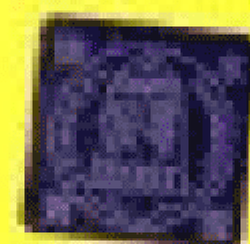
赠送4小时本书实例讲解视频

赠送800页电子书

提供280页PPT文档

CD-ROM

强锋科技　赵辉　编著



清华大学出版社

网站开发指南

HTML+CSS网页设计指南

强锋科技

赵　辉　编著

清华大学出版社

北　京

内 容 简 介

本书主要介绍了页面前端技术的开发，即常说的“HTML+CSS+JavaScript”。现在的 Web 设计理念较之于前两年，已经发生了很大的改变，CSS 比传统的属性标签使页面变得更漂亮，而将来，页面中要求的不仅仅是美观，甚至是越来越多的互动功能，JavaScript 即是目前最流行的页面脚本语言。

2009 年，互联网巨头 Google 公司在全球推广 Google Chrome 浏览器，正如 Chrome 浏览器本身：“今天，我们日常生活所用的互联网不再仅仅是网页，而是应用程序。”本书针对当前网页的开发，详细介绍了 CSS 的使用法则，不仅如此，本书同时注重于 JavaScript 的使用，目的就在于不仅着眼于今天最流行、最成熟的技术，更要把握明天页面开发的趋势。

本书从最基本的 HTML 标签讲起，适用于 Web 开发的初学者，而对于有一定 Web 前端开发基础的读者，也有一定的参考意义。

图书在版编目（CIP）数据

HTML+CSS 网页设计指南/赵辉编著．—北京：清华大学出版社，2010.1
（网站开发指南）

ISBN 978-7-302-21302-4

I. H…　II.赵…　III. ①超文本标记语言，HTML、XHTML-主页制作-程序设计　②主页制作-软件工具，CSS

IV. TP312　TP393. 092

中国版本图书馆 CIP 数据核字（2009）第 181357 号

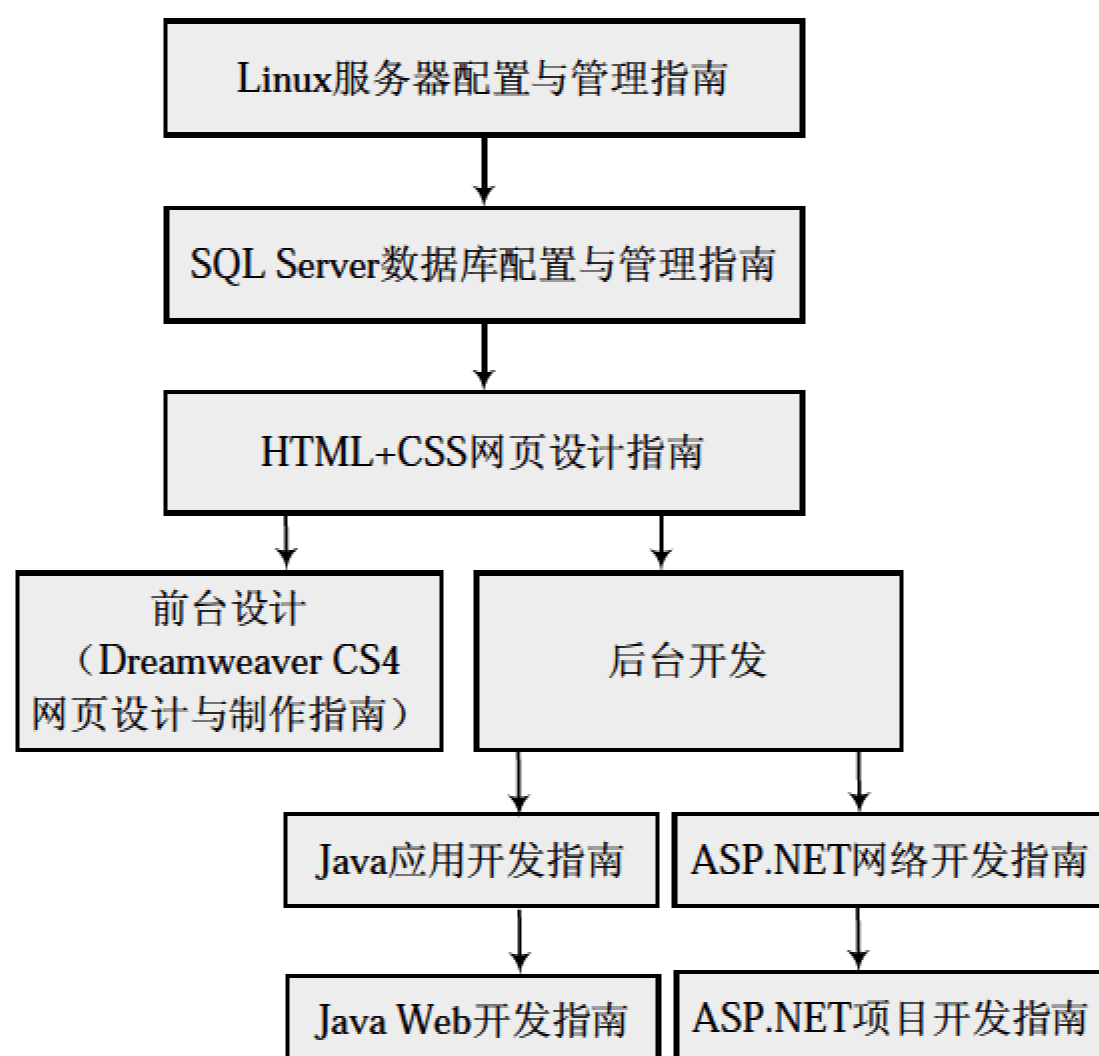
责任编辑：朱英彪　张丽萍
封面设计：张　岩
版式设计：王世月
责任校对：王　云
责任印制：
出版发行：清华大学出版社　　**地　　址：**北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编：**100084
社 总 机：010-62770175　　**邮　　购：**010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：203×260　**印　　张：**24.75　**字　　数：**659 千字
（附光盘 1 张）
版　　次：2010 年 1 月第 1 版　**印　　次：**2010 年 1 月第 1 次印刷
印　　数：1～4000
定　　价：42.80 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：010-62770177 转 3103　　产品编号：034200-01

前　　言

随着网站技术的进一步发展，各个部门对网站开发技术的要求日益提高，综观人才市场，各企事业单位对网站开发工作人员的需求也大大增加。但是网站建设作为一项综合性的技能，对很多计算机技术都有着很高的要求。网站开发工作包括市场需求研究、网站策划、网页平面设计、网站程序开发、数据库设计以及网站的推广运作等，可以系统掌握这些知识的网络工程师相对较少。

如此诸多方面的知识，使得很多初学者往往都会感到十分困惑，不知道各项技术之间的关系。本套丛书正是由此而来，并完美地解决了这个问题——为广大读者学习网站开发技术提供一个完整的学习方案。丛书的组织结构如下：



本书是其中的一本，用以帮助读者掌握HTML和CSS的用法。HTML和CSS是网络的公共语言，所有的网站都是通过它们来与客户端沟通的。互联网从1969年诞生至今，已经走过了40个年头。起初，互联网只适合于计算机专家、工程师和科学家使用。到了20世纪末，互联网有了较大的发展，但作为时髦一族的新兴学科、一个新兴名词，并未被大部分人了解和接受。进入21世纪，互联网的发展突飞猛进，如今已经成为现代人生活中的必需品。

十多年前，笔者第一次使用Frontpage制作出了自己的第一个页面，当时现在看来那个页面真的很糟糕，它只是一些颜色块、一个标题和一张飞入页面的图像，但当时将它展示给朋友时，那种通过自己的作品而带给朋友之间的快乐至今仍然记忆犹新。现在，笔者作为一个前端页面设计师，享受设计快乐的同时，也令这个技能成为自己的一份职业，希望通过本书可以将这份对页面制作的热爱和读者一起分享。

本书的特点

本书深入浅出地讲解了网页的概念和制作网页的方法，以及目前流行的各种制作网页的技术和常用的网页制作工具。每章的例子都是专门针对一项技术而设计的，而且每个例子中都考虑了这项技术的实用性、趣味性，并将这种特色融于例子中。

笔者精心编写了这本书，其目的就在于能够系统、全面地介绍制作前端页面的技术。简而言之，本书的宗旨是使读者可以做出自己心目中的网页样式，帮助读者迅速掌握制作页面的技巧，由一个初级入门者，最终成长为一位成熟、专业的页面设计人员。

本书的特点主要体现在以下几个方面：

- 本书的编排采用循序渐进的方式，适合初、中级读者逐步掌握制作页面的基本方法，以及设计页面和管理页面的精髓。
- 本书结合网页制作技术十多年的发展过程，介绍了早期的技术，重点介绍了目前流行的技术，同时也介绍了未来技术的发展趋势，理解这个发展过程，会帮助读者更好地掌握制作页面的设计思路和精髓。
- 本书在介绍各种页面制作技术时，采用了许多风趣易懂的例子，这些例子具备完整的代码，在介绍理论知识的同时，更注重全书的实用性。因此，读者自己进行实践和演练时就不会对理论感到困惑。本书的所有例子、源代码和各种免费工具都附在随书光盘中，方便读者使用。
- 本书除介绍页面制作外，还补充了大量相关的计算机基础知识，如图像格式、颜色模式等，这些都是很重要的基础知识，它们不仅能帮助读者深入理解网页制作，对图像制作、动画制作等领域也会起到融会贯通的效用。
- 本书结合笔者多年的制作经验，在每章结束时都会给出一个综合实例，以供读者学习和巩固所学知识。
- 本书附带一张多媒体光盘，书中所有的实例及操作在光盘中均有源文件。

本书的内容安排

本书分为 4 篇，共 19 章，从制作页面的基本概念讲起，再进一步介绍一个页面的设计过程所经历的各个阶段，然后结合目前成熟的制作方法和理念，讲解如何进行页面的设计制作，最后通过 4 个综合案例让读者加深印象。

第 1 篇（第 1 章～第 6 章）　页面制作入门篇：讲述了页面制作的基础知识，包括互联网的起源和发展、页面的基础概念、页面的运行原理，以及制作页面的基础知识。读者通过这 6 章的基础知识来了解什么是页面，如何在页面中添加基本的元素，以及重要的互联网中的属性概念，如计算机中的距离、颜色、图像等描述方式。通过对这部分的学习，读者可以制作简单的页面。

第 2 篇（第 7 章～第 11 章）　页面制作提高篇：讲述了页面制作的成熟技法和专业制作，让读者从制作简单页面阶段进阶到专业的制作页面阶段。通过“CSS+DIV”的学习，让读者可以了解如何使用技巧做好页面的布局设计，如何使用样式表来让自己的设计变得游刃有余，使天马行空的设计理念成为现实。

第 3 篇（第 12 章～第 15 章）　动感页面篇：讲述了页面制作的互动作用，是读者从普通页面制作人员迈向高手之路的铺垫。这部分介绍了页面和用户互动的原理以及如何令页面和用户实现互动；

介绍了什么是脚本语言，通过脚本语言读者可以制作基础的动态页面。

第 4 篇（第 16 章～第 19 章）　页面实战篇：介绍了制作页面的实用工具，通过 4 个综合案例，展现了一个页面如何从一个想法开始最后成为一个完整的页面。

适合阅读本书的读者

- 希望进入网站制作领域的新手。
- 希望系统学习网页制作的页面制作人员。
- 具备一定的页面基础理论但缺乏实践操作的人员。
- 对页面设计有着浓厚兴趣的同仁。
- HTML 语言爱好者。
- 网站管理人员。
- 页面样式设计人员。

本书作者

本书由赵辉组织编写，其他参与编写、资料整理、代码调试的人员还有陈刚、宫磊、谷原野、黄其武、李修花、李延琨、林家昌、刘林建、孟富贵、彭自强、孙雪明、王世平、文明、徐增年、银森骑、张家磊、张瑾瑜、周伟杰、朱玲、陈杰、陈冠军、张金霞、张昆和尹继平，在此一并表示感谢。

编　者

目 录

第 1 篇 页面制作入门篇

第 2 篇 页面制作提高篇

第 3 篇 动感页面篇

第 4 篇　页面实战篇

第 1 篇　页面制作入门篇



第 1 章　了 解 网 页

15 年前，“网上冲浪”还是一个很时尚的名词，但随着互联网的飞速发展，时至今日，使用网络已经成为人们生活中的一部分。Web 1.0 时代，人们只是在网页上浏览信息，而在现在所处的 Web 2.0 时代里，人们可以在网页上和朋友讨论话题、听音乐、看电影、进行电子商务的操作。《纽约时报》专栏作家托马斯·弗里曼在他的《世界是平的》一书中说到：“2000 年世界进入了一个新纪元：全球化 3.0。世界从小缩成微小，竞赛场地铲平了。”

本章将介绍一些与互联网相关的常用技术解释，以及制作网页时通常需要涉及的领域和考虑的问题等。本章的知识点很多但并不难理解，读者千万不要被那些可怕的名词或代码给吓住。虽然 HTML（Hypertext Marked Language，超文本标记语言）标签很多，但是在找到规律后也就千篇一律了。好比去大卖场买菜，第一次进去真的很无奈，因为根本分不清蔬菜的种类，至少在笔者眼里，感觉都是绿色的“草”。但是多去几次后就可以很轻松地做这件事了。事实上，学习 HTML 不会比买菜难。本章的知识点如下。

- ❑ 网页的概念和分类：了解网页的概念，区别静态网页和动态网页的不同。
- ❑ 设计网页的原则和工具：了解开发网页的常用工具。
- ❑ 网页浏览器的工作原理：知道网页浏览器的工作原理，并了解 3 种常用的浏览器。
- ❑ HTML、XML（The Extensible Markup Language，可扩展标识语言）和 XHTML（The Extensible Hypertext Markup Language，可扩展超文本标识语言）的概念，区别这 3 个概念的异同点。
- ❑ HTML 应用：通过本章最后的实例，演示一个简单的 HTML 页面的开发步骤。

1.1　什么是网页

21 世纪是信息化的时代，互联网使用者可以在 Internet 上通过网页浏览信息，如新闻、图片等；也可以通过互联网发布信息，如招聘信息、各种广告；或者是追赶潮流，加入现今十分流行的博客一族。然而这些都离不开互联网的这扇窗户——网页。本节将介绍网页的相关知识。

1.1.1　网页的概述

网页其实是在这个世界的某一个地方某一台计算机上的一个文件，通过互联网，也就是 Internet 将两个不同的地址相连，把人们的信息传达到网络世界的各个角落。人们通过互联网，可以在世界的任何一个地方互相交流沟通。

什么是互联网？互联网是一种概念，一个虚拟的东西。人们可以通过浏览器浏览新浪、百度等页面，但不会有人说“浏览互联网”。互联网指的正是把所有网页链接在一起的巨大信息交流形式，它基于很多的协议来体现出它的表现形式。1989 年欧洲粒子物理实验室的科学家们提出了一个分类互联

网信息的协议。这个协议极大地推动了互联网的发展和普及，后来它有了一个十分响亮的名字 WWW（World Wide Web），中文又称万维网（Wan Wei Wang）。可以认为，从 20 世纪 90 年代开始，互联网进入了 Web 1.0 的时代。

早期的 Web 1.0 时代，大部分网页只有文字、图像信息可以浏览，这个时候最典型的互联网标志就是门户网站，如新浪、搜狐。从 2001 年开始，人们认为互联网开始进入 Web 2.0 时代，这时的网页可以包含动画、音频、视频，也可以在网页中进行交流、上传文件，使用完全交互式的程序。互联网开始更注重个人化的网络服务，任何使用网络的人，都可以参与到网页的制作中。

笼统地说，网页主要由结构、表现和行为 3 部分组成。对应的标准也分为 3 类，其中大部分都是由 W3C（World Wide Web 协会）所制定，这其中就包括 HTML 和 CSS（Cascading Style Sheets，即层叠样式表）。对于初学者来说，了解网页的制作，一定要分清楚 HTML 和 CSS 的不同作用。



说明：HTML 标准基于语义学，通过标签来应用语义的过程，使用起来好像是打手语，做一个手势，去告诉对方代表了什么意思。不同的是，这里的手势换成了标签。

例如，使用<table>，这是一个表格标签，意思是“将在这里放入一个表格”。那么这个表格该如何表现在浏览者的面前呢？如它的颜色、边框粗细等。而 CSS 的出现为设计者解决了这些问题，如图 1.1 和图 1.2 所示为不同样式的表格。



图 1.1　粗边框的表格



图 1.2　细边框的表格

图 1.1 和图 1.2 都是有着一个 3×3 表格的页面。这样的描述，如同 HTML 语言所表达的含义，体现出页面上的内容，而在浏览器中最后的显示效果是完全不同外表的两个表格。图 1.1 表格的边框较粗，黑色；图 1.2 表格边框较细，红色。这是因为它们使用了不同的 CSS 样式表。

所以通俗地说，HTML 表现了页面的结构，而 CSS 修饰了页面中的这些内容。如果把制作网页比作一个人在设计一间屋子，那么 HTML 语言的作用是用来明确这个屋子内要放入哪些家具，或者是床、书柜、椅子等。而 CSS 的作用就是改变这些家具的样式，对应的如床的样式、书柜的样式、椅子的样式等。

1.1.2　静态网页

在网站设计中，纯粹的 HTML 格式网页通常被称为“静态网页”，早期的网站一般都是由静态网页组成的。静态网页的特点是这个网页不论何时何地浏览，都将显示相同的形式和内容，且仅能供浏

览，无法与网站进行互动。也就是说，无法提供信息给网站，让网站响应使用者的需求。

【本节示例参考：资料光盘\第 1 章\1-1　静态页面.html】

【实例 1-1】一个静态页面的代码展示如下：

程序 1.1　静态页面.html

```
01  <html>
02   <head>
03       <title>静态页面</title>
04   </head>
05   <body>
06       <h1>这是一个静态页面</h1>
07       <h3>您只能浏览，不能进行交互。</h3>
08   </body>
09  </html>
```

【运行程序】浏览该页面，结果如图 1.3 所示，它只能浏览而不能被交互。

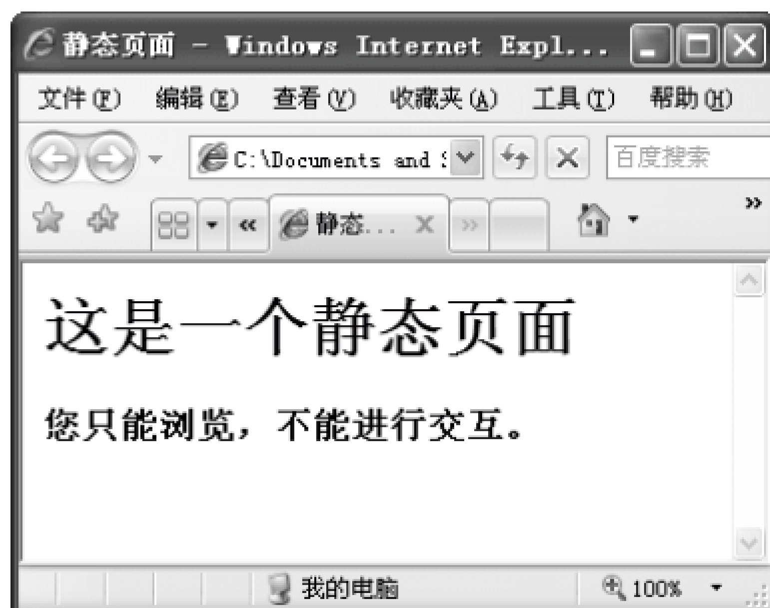


图 1.3　静态页面示例

【深入学习】静态页面使用 HTML 编写，通常扩展名为.htm、.html、.shtml、.xml 等。静态网页只能单纯地在网页中展示文字与图片，听起来似乎功能简单，但它是所有网页的基础要素，其重要性不言而喻。

在静态网页中，整个网页的主要结构与网页的显示控制都必须利用 HTML 实现。在 HTML 格式的网页上，可以出现各种动态的效果，如.GIF 格式的动画、Flash、滚动字幕等。这些“动态效果”只是视觉上的，而动态网页是不同的概念。



注意：本书中大部分的实例都是静态页面。

1.1.3　动态网页

动态网页是与静态网页相对应的，指网页的内容可以根据某种条件的改变而自动改变。如腾讯公司的 Qzone 空间里，常常会有一些使用者嵌入一个小小的计数器功能，当有人单击设计者的网页时，计数器的值会自动增加。这个计数器就是动态的。再比如，目前网络流行的论坛、社区网，其中用户

的注册页面，当用户输入正确的用户名和密码后会成功登录，如果输入的用户名或密码错误，页面会提示用户错误信息。这也是典型的动态页面。

同时，与静态网页的后缀不同，动态网页是以.asp、.jsp、.php、.perl、.cgi 等形式为后缀，并且在动态网页网址中有一个标志性的符号“?”。一个典型的动态网页的 URL（Uniform Resource Locator，即统一资源定位器）形式为：http://www.sina.cn/ip/index.asp?id=1。那么，动态网页与网页上的各种动画、滚动字幕等视觉上的动态效果为什么不是一个概念呢？

动态网页可以是纯文字内容的，也可以是包含各种动画的内容。这些只是网页具体内容的表现形式，并不是动态技术生成的页面，它不能根据用户的要求来更新页面。而一个网页，无论是否具有动态效果，除非是采用动态网站技术生成的网页，才可以称为动态网页。



注意：分清静态页面和动态页面的区别很重要，概念清晰才能进一步理解哪些是 HTML 语言可以做到的，哪些是不允许的，避免在学习的过程中钻牛角尖。

在了解了动态网页的概念之后，下面通过一个动态网页的例子来说明动态页面是如何和用户互动的。如图 1.4 所示是一个动态页面的第一个页面。当填入一个已注册用户，如用户名为 appleing，密码为 1234567 时，系统检查到用户名和密码是正确的，即跳转到用户页面，如图 1.5 所示为一个用户页面。



注意：这种功能是静态页面无法做到的。或者说，仅仅依靠 HTML 代码是无法达到这种效果的。



图 1.4　userLogin.jsp 页面



图 1.5　用户页面

图 1.5 所示的页面可以显示该用户使用的功能，包括用户邮件服务和用户短消息服务。这种针对使用者的设计是动态页面典型的功能标志。而如果用户输入错误的用户名和密码，如用户名为 bupt，密码为 bupt，则系统会检查到用户名和密码是错误的，会跳出错误信息，显示登录失败。如图 1.6 所示为登录失败的页面。



图 1.6　登录报错页面

以上是一个典型的动态页面的例子，用户输入不同的信息，则页面显示不同的结果。动态页面比静态页面显得更智能、更人性化，自然也是互联网发展的趋势。而理解 HTML 页面是学习动态页面的必备基础知识。

1.1.4　开发动态页面和静态页面的联系

早期的动态网页主要采用 CGI（Common Gateway Interface）技术，是 HTTP 服务器与用户或其他机器上的程序进行“交谈”的一种工具，其程序须运行在网络服务器上，可以使用多种不同的程序语言（如 Visual Basic、Delphi 或 C/C++等）编写适合的 CGI 程序。

当用户在页面上留言，输入一段信息，接着单击“确定”按钮时，都是工作在客户端。而接下来浏览器会将用户的信息传递到 CGI 程序，于是 CGI 程序在服务器上按照预定的方法进行处理。虽然 CGI 技术已经应用很长时间，而且功能比较强大，但由于其有编程困难、效率低下、修改复杂等缺点而逐渐被新技术取代。目前常用的新技术有 3 种，这 3 种技术在制作网页上各有特色，但都在发展中。在互联网领域中，这 3 种技术的格局像三国分立的局面一样，都占有市场的一席之地，它们就是 PHP 脚本语言、ASP 脚本语言和 JSP 脚本语言。

1. PHP 脚本语言

PHP 即 Hypertext Preprocessor（超文本预处理器），是目前在 Internet 上应用较为广泛的脚本语言，其语法借鉴了 C、Java、Perl 等语言。PHP 对编程能力的要求不高，只需少量的编程知识就可以使用 PHP 建立一个交互的 Web 站点。

PHP 与 HTML 语言具有非常好的兼容性，因此它与 HTML 可以结合使用从而更好地实现页面控制，即直接在脚本代码中加入 HTML 标签，或者在 HTML 标签中加入脚本代码。PHP 提供了标准的数据库接口使得连接数据库比较方便，另外还具有扩展性强、可以进行面向对象编程等特点。

2. ASP 脚本语言

ASP 即 Active Server Pages，是微软开发的一种语言，它本身没有专门的编程语言，而是允许用户使用许多已有的脚本语言编写 ASP 的应用程序。ASP 的工作方式是在 Web 服务器端运行，运行后再将运行结果以 HTML 格式传送至客户端的浏览器。正因为如此，ASP 要比一般的脚本语言安全得多。

ASP 可以包含 HTML 标签，也可以直接存取数据库及使用可扩充的 ActiveX 控件，因此 ASP 的程序编写比 HTML 更方便且更有灵活性。但 ASP 技术基本上是局限于微软的操作系统平台之上，主要工作环境是微软的 IIS 应用程序结构，而且 ActiveX 对象具有平台特性，所以 ASP 技术跨平台性不是很

好，即实现在跨平台 Web 服务器上工作不是很容易。

3．JSP 脚本语言

JSP 即 Java Server Pages，它是由 Sun Microsystem 推出的，是以 Java Servlet 和整个 Java 体系的 Web 开发技术为基础。JSP 和 ASP 在技术方面有许多相似之处，不过 ASP 一般只应用于微软的平台上，而 JSP 则可以在大多数的服务器上运行，而且采用 JSP 技术开发的应用程序比采用 ASP 开发的应用程序更具有可维护性和可管理性，所以，JSP 被业界认为是未来最有发展前途的动态网站技术。

这 3 种技术目前是制作动态页面的主流，而且在未来一段时间内，PHP、ASP、JSP 也会在竞争中共存。对微软服务器较熟悉的程序员采用 ASP 技术更容易得心应手。对于 Linux 的爱好者来说，采用 PHP 技术是比较合适的选择。对可维护性和可管理性要求较高的设计者，适合使用 JSP 技术，尤其是在构建大型网站的应用中。



注意：动态页面的制作相对 HTML 页面复杂很多，了解动态页面的制作方式和 HTML 页面的关联，能够帮助初学者学习理解静态页面。

事实上，动态页面的设计是离不开 HTML 的，每一个动态网页开发者都必须掌握 HTML 语言。在实际应用的网页中，动态网站中包含大量的 HTML 代码，合理结合静态页面和动态页面可以使网页设计更加灵活。

此外，设计者应明白，动态网站不一定比静态网页更好，动态网站的交互性可能带来安全隐患，而且动态网站的信息均是从数据库中读取，当负荷过大时可能造成网站崩溃。或者动态网站对于搜索引擎不是很友好，在一些搜索页面中不容易被查找，这样会影响网站的推广。

因此，无论是从 HTML 在网页设计中的基础地位还是其优点来说，熟练掌握 HTML 对于网页设计是很重要的基础。在这个基础上再向动态页面设计的道路前进时，会更容易掌握动态页面的使用技术。

1.2　开发网页的工具

如果说制作网页的设计者是一个画家，那么开发程序的工具就相当于画家手中的画笔和颜料。这些工具能为设计者编写、调试、运行代码时提供一个方便的环境。在开发网页时也有属于它的开发工具和为这种语言量身定做的开发工具，如记事本、Dreamweaver 等工具可以用来开发 PHP、ASP 和 JSP 中的任何一种程序。

1.2.1　HTML 页面的开发工具

HTML 语言作为一种语义派生出来的语言，最常见的有 3 种开发工具，分别是记事本、Dreamweaver 和 Frontpage。

1．记事本

记事本是 Windows 系统自带的简单的文本编辑软件，但由于大部分代码都是纯文本的，所以记事

本可以编写任何网页。不过对于稍大型的网页需要编辑大量代码时，记事本就显得不那么适合了，但对于初学者来说，记事本是较好的练习工具。

2．Dreamweaver 开发工具

Dreamweaver 是 MACROMEDIA 公司开发的集网页制作和管理网站于一身的网页编辑器，它是一款专业网页设计师的视觉化网页开发工具，利用它可以制作出跨越平台限制和跨越浏览器限制的充满动感的网页。

Dreamweaver 最大的特点是所见即所得。可以使用它的网站地图快速制作网站雏形，设计、更新和重组网页。此外，Dreamweaver 可以自动生成源代码，大大提高了网页开发人员的工作效率。但是 Dreamweaver 也有其自身的缺点，在一些复杂的网页中，难以精确达到与浏览器完全一致的显示效果。同时其产生的代码效率比较低。

Dreamweaver 是一款可视化编辑工具，如图 1.7 所示为 Dreamweaver 工作状态的操作界面。Dreamweaver 不仅支持静态页面的编写，而且还支持 PHP、ASP、JSP 等动态网页的编写与调试。对于网页设计初学者来说，Dreamweaver 是一款比较好的入门软件，即使对 HTML 不太熟悉，也能做出漂亮的网页。

Dreamweaver 的工作界面是分割成一块块的窗口，这些工具窗口可以按照用户自定义的样式自由设定。从图 1.7 中可以看到，最引人注意的两个工作区域分别是代码区和预览区。



说明：关于 Dreamweaver 的进一步介绍可以参考第 15 章。

3．Frontpage 开发工具

Frontpage 是微软公司发布的一款入门级网页制作工具，是微软 Office 组件的一部分。相比于 Dreamweaver，Dreamweaver 的直观性和高效性是 Frontpage 无法比拟的，而且在功能拓展方面 Frontpage 也少于 Dreamweaver。因此，2006 年 Frontpage 停止了发售，但并不能因此否认这是一款失败的制作软件，因为它依然受到众多用户的欢迎。如图 1.8 所示为 Frontpage 的操作界面。

由于 Dreamweaver 的适用性目前已经超过了 Frontpage，因此在大多数情况下，设计人员的第一选择已经是 Dreamweaver。



图 1.7　Dreamweaver 的操作界面

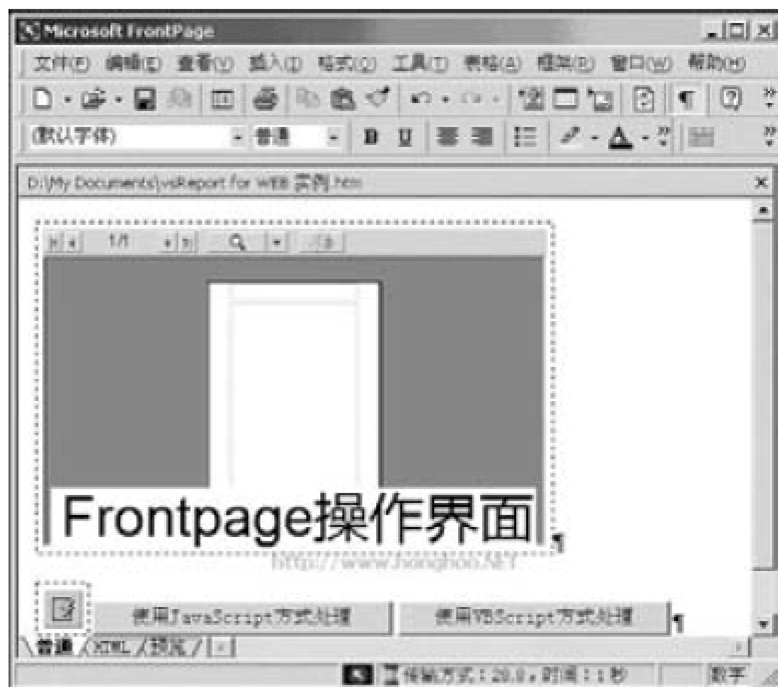


图 1.8　Frontpage 的操作界面

1.2.2　动态页面的开发工具

动态页面的开发工具根据开发语言的不同而不同。本节主要介绍常见的 3 种开发语言：PHP、ASP、JSP 的开发工具。

1. PHP 开发工具

PHP 作为一种开放型的语言，其开发工具并没有标准的规定，可以用很多工具进行开发（包括记事本），也可以用 ZDE 等工具进行开发，只要语法正确即可。比较好的是 ZDE 和 PHPED 开发工具。

ZDE（Zend Development Environment）是 ZEND 公司推出的一款集成开发平台。ZDE 是用 Java 语言编写的，其同样具有多平台性。

PHPED 是由 NUSPHERE 公司推出的，它提供的功能最全，同样具有语法加亮、函数补全和工程管理等功能。除此之外，还有自动代码补全、可视化的数据库管理、常见 HTML 标签集、支持插件等功能。同时它还内置 DAV、CVS、FTP、WEBSERVER、DEBUGGER、JS 代码列表。



说明：对于 HTML 页面的开发者来说，可以不需要了解这两种开发软件，它们主要针对开发 PHP 页面。

2. ASP 开发工具

Visual InterDev 是 Microsoft 公司所开发的 ASP 开发工具，是一款可视化工具，可以对 ASP 代码进行颜色识别、自动代码提示。Visual InterDev 是为程序员设计的网页开发工具，而 Microsoft Frontpage 是针对非程序员的编辑工具。Microsoft Frontpage 是 Microsoft Office 中的一部分，Visual InterDev 则是 Microsoft Visual Tools 中的一部分，其外观和工作模式均与其他 Microsoft 可视开发工具类似，如 Microsoft Visual C++。

3. JSP 开发工具

MyEclipse+Struts 是比较流行的 JSP 开发工具。Struts 最早是作为 Apache Jakarta 项目的组成部分，项目的创立者希望通过对该项目的研究，改进和提高 Java Server Pages、Servlet、标签库以及面向对象的技术水准。Struts 这个名字来源于在建筑和旧式飞机中使用的支持金属架，这个框架之所以称为 Struts，是为了提醒人们记住那些支撑房屋、建筑、桥梁，甚至在踩高跷时的基础支撑。

这也是一个解释 Struts 在开发 Web 应用程序中所扮演角色的精彩描述。当建立一个物理建筑时，建筑工程师使用支柱为建筑的每一层提供支持。使用 JSP 开发工具开发简单、维护方便，而且便于安装插件，可以与 Spring、Hibernate 等结合使用，满足开发者的需要。



说明：这些都是动态页面的开发环境，并不是静态页面的范畴，故这里只做一个简要的介绍。

1.3 使用网页浏览器

网页浏览器是显示网页服务器或档案系统内的文件，并让用户与这些文件互动的一种软件。它用来显示在万维网或局部局域网络等内的文字、影像及其他资讯。浏览器就是设计者的画廊，设计者把网页放在这里展示给用户。

1.3.1 网页浏览器的工作原理

Windows 系统中自带了 IE 浏览器，普通用户在使用它浏览网页时，很多时候都忽视了自己在使用浏览器。对于一个页面设计者来说，了解浏览器的原理可以令设计者找到适合的途径把网页展示给用户。

那么用户是如何使用浏览器浏览网页的呢？WWW 是一种采用 B/S（Browser/Server）的结构，即浏览器和服务器结构。它是随着 Internet 技术的兴起，对 C/S 结构的一种变化或改进的结构。在这种结构下，用户工作界面是通过 WWW 浏览器来实现的，主要事务逻辑在服务器端（Server）实现，很少部分事务逻辑在前端（Browser）实现。这样的好处是大大简化了客户端的计算机载荷，减轻了系统维护与升级的成本和工作量，降低了用户的总体成本。因此，用户只需要安装浏览器即可浏览页面，不需要知道服务器端使用什么操作系统或服务器端怎么处理浏览器发出的请求，可以方便查看自己想看到的内容。

浏览器的工作原理可以分以下几步来理解：

（1）浏览器通过 HTML 表单或超链接请求指向一个应用程序的 URL。

（2）服务器收发到用户的请求。

（3）服务器执行已接受创建的指定应用程序。

（4）应用程序通常是基于用户输入的内容，执行所需要的操作。

（5）应用程序把结果格式化为网络服务器和浏览器能够理解的文档，即我们所说的 HTML 网页。

（6）网络服务器最后将结果返回到浏览器中。

如图 1.9 所示为浏览器的工作原理流程图。

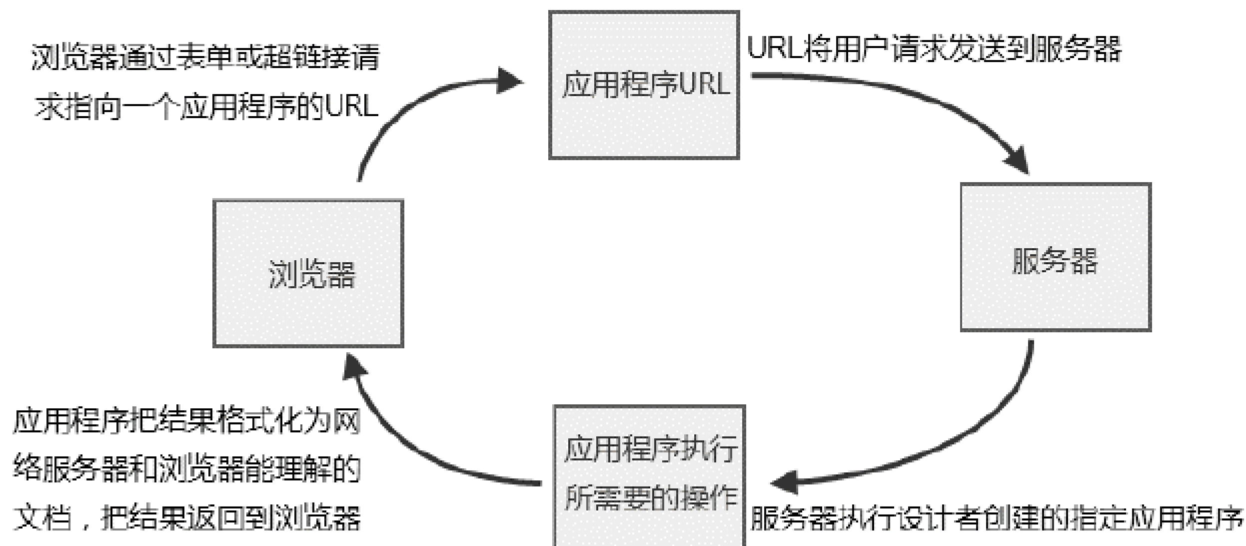


图 1.9 浏览器的工作原理流程图

图 1.9 是一个从用户在浏览器输入网址到浏览器显示页面的工作过程。WWW 的基础是 HTTP 协议，Web 浏览器就是用于通过 URL 来获取并显示 Web 网页的一种软件工具。URL 用于指定要取得的 Internet 上资源的位置与方式。

因此，并不是所有浏览器都支持 HTML 语言标签。在这种情况下，需要在 HTML 中添入声明作用的代码，相关的知识点在本书后面的章节中将会详细介绍。

1.3.2　常用的两种浏览器

目前互联网上最常用的有两种浏览器，分别是 Internet Explorer 和 Mozilla Firefox，它们基于不同的内核，各有特色，难分伯仲。

Internet Explorer 简称 IE 或 MSIE，是微软公司推出的一款网页浏览器。自从 2004 年上市以来，虽然目前丢失了一部分市场占有率，但 IE 浏览器依然是目前使用最广泛的网页浏览器。它被捆绑作为所有新版本的 Windows 操作系统中的默认浏览器，如图 1.10 所示。

Mozilla Firefox 是由 Mozilla 公司开发的一个自由的、开放源码的浏览器，是一款可以与 IE 系列浏览器一争高低的浏览器。这款浏览器有效地支持 W3C 各项国际标准，有 Windows、Linux 等多个平台的版本，其特点是轻便、安全、分栏浏览，第三方扩展非常多，可以极大地提高浏览的乐趣。

除此之外，由于是开源组织维护 Firefox，所以它对错误的修正是极为迅速的，新功能的实现也都是基于用户的要求。由于 Firefox 本身的优秀特效，吸引了许多人义务为它宣传，不断扩大它的影响力，现在它的市场占有率正不断提高，开始撼动 IE 的垄断地位。如图 1.11 所示为 Firefox 浏览器。



图 1.10　IE 8 浏览器



图 1.11　Firefox 浏览器

在这两种浏览器中，IE 浏览器使用最简单，占用计算机资源少，而 Firefox、Maxthon 以其个性化设置也吸引了众多爱好者，其缺点是占用了较多的计算机资源，容易造成计算机使用缓慢。用户在使用时可以根据需求选择不同的浏览器查看设计的页面效果。

1.4　HTML、XML 和 XHTML 语言

最初 HTML 仅允许研究人员以一种非常高效的方式在互联网上共享信息，但是当网页浏览器这一工具变得复杂、多样化之后，设计人员可以在页面中放入更多的不同形式的文件，如图像、音频。因此，网页开发人员开始为 HTML 语言加入了更多的标签，而使本来设计简单的 HTML 开始变得丰富。但其带来不利的结果是使 HTML 使用规则变得混乱，这时就需要新的规则来约束网页的定义，使同样内容的网页在不同浏览器中显示一样的结果，XML 在这个方向发挥着重要的作用。

1.4.1 超文本标记语言 HTML

HTML 是一种用来制作超文本文档的简单标记语言，用其编写的超文本文档称为 HTML 文档，它能独立于各种操作系统平台。

1990 年以来，HTML 一直被用作 World Wide Web 的标准表示语言，用于描述 Homepage 的格式设计以及它与 WWW 上其他 Homepage 的连接信息。作为一种最为基础的语言，人们使用它描述的文件需要通过 WWW 浏览器显示出效果。

所谓超文本，因为它可以加入图片、声音、动画、多媒体等内容，不仅如此，它还可以从一个文件跳转到另一个文件，与世界各地主机的文件连接。HTML 的作用是用来展示页面的表现形式，如页面的布局、页面的颜色、页面中的内容等。

HTML 已经成为全球信息网的基础，它提供标准化的方法将信息格式化并经由 Internet 传送给全世界的使用者。HTML 为人们的传送和接收信息带来了革命性的变化，但是 HTML 主要是被设计为资料显示之用。

1.4.2 可扩展标识语言 XML

HTML 的焦点几乎完全集中在信息应如何显示上，而不是信息的内容及它的结构上，所以新规则中需要引入 XML。XML 和 HTML 的作用区别很大。

XML 目前推荐遵循的是 W3C 于新世纪初发布的 XML 1.0。和 HTML 一样，XML 也来自于 SGML，它最初设计的目的是补充 HTML 的不足，以强大的扩展性满足网络信息发布的需要，后来逐渐用于网络数据的转换和描述。它是一种能定义其他语言的语言，目前在网站信息传递中常用的 RSS 就是典型的 XML 应用。如果说 HTML 是用来设计页面的布局和视觉效果，那么 XML 是用来描述页面的数据形式和结构。

需要注意的是，XML 并不是标记语言，所以它不是 HTML 的升级。它更多的作用是对 HTML 做一些其他功能的补充，而仅使用 XML 是无法写出页面的。XML 文档代码如下：

```
01  <?XML version="1.0"?>
02  <myfile>
03  <title>我的文档 </title>
04  <author>Depp</author>
05  <email>depp59@gmail.com</email>
06  <date>20090109</date>
07  </myfile>
```

第 1 行是 XML 文档的声明，这和 HTML 文档很像。第 2 行是代码的根元素，类似 HTML 中的<html>开头标记（参考本章程序 1.1）。再下来的标记是描述这段内容的标签，注意这些标签名称都是可以随意定义的。如果读者愿意，甚至可以用中文写成如“<标题>我的文档 </标题>”这样，而这在 HTML 中是不可以的。



说明：XML 数据是使用脚本实现 HTML 中调用和互动的，所以使用 XML，必须掌握一门脚本语言的使用技巧。

1.4.3　可扩展超文本标识语言 XHTML

XHTML 是 2000 年 W3C 公布发行的，它不需要编译即可直接由浏览器执行（属于浏览器解释型语言），是一种增强了的 HTML。它的可扩展性和灵活性将适应未来网络应用更多的需求，是基于 XML 的应用。可以说 XHTML 就是 HTML 的一个升级版本，它们之间的区别很小，有时在使用上很难分清它们之间的界线。

XML 虽然数据转换能力强大，完全可以替代 HTML，但面对成千上万已有的站点，直接采用 XML 还为时过早。因此开发者在 HTML 4.0 的基础上，用 XML 的规则对其进行一些扩展，由此得到了 XHTML。简单地说，建立 XHTML 的目的是为了实现 HTML 向 XML 的过渡。

正是因为这样，XHTML 文档必须使用小写，因为 XML 是大小写敏感的，如<H1>和<h1>是不同的标签。此外，XHTML 中要求标签必须有始有终。当然从页面设计者的角度来说，无论 HTML 还是 XHTML 的代码写法都是正确的。为了更严谨的写法，使用 XHTML 的写法要求未尝不可。

1.5　编写一个简单的页面

本实例的目的在于展示一个页面制作的完整代码，其中涉及一个重要的概念——网页的超链接。单独的页面如同一本书中的一页，把所有的页面链接在一起，如同是把所有的书页装订在一起使之称为书，而所有的页面链接在一起称之为网站。

超链接是网页独有的特色技术，如在下面的实例中，单击“这是我的第一个网页，我喜欢 HTML!您可以单击这里”链接，页面会跳转到搜狐主页。首先打开记事本，输入程序 1.2 所示的页面源码。

【本节示例参考：资料光盘\第 1 章\1-2　编写一个简单页面.html】

【实例 1-2】编写一个简单页面，其代码展示如下：

程序 1.2　编写一个简单页面.html

```
01  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03  <html xmlns="http://www.w3.org/1999/xhtml">
04      <head>
05          <meta http-equiv="Content-Type" content="text/html; charset=GBK">
06          <title>Hello！欢迎使用 HTML</title>
07                      </head>
08      <body>
09          <p><H1>这是我的第一个网页,我喜欢 HTML!您可以单击这里
10                  <H2><a href="http://www.sohu.com ">
11                          浏览进入搜狐，查找您感兴趣的东西!
12                      </a>
13          </p>
14      </body>
15  </html>
```

这些都是 HTML 标签



注意：<a herf="">的功能是链接其他页面，本书会在后面的章节中详细介绍。

【运行程序】浏览该页面，结果如图 1.12 和图 1.13 所示。

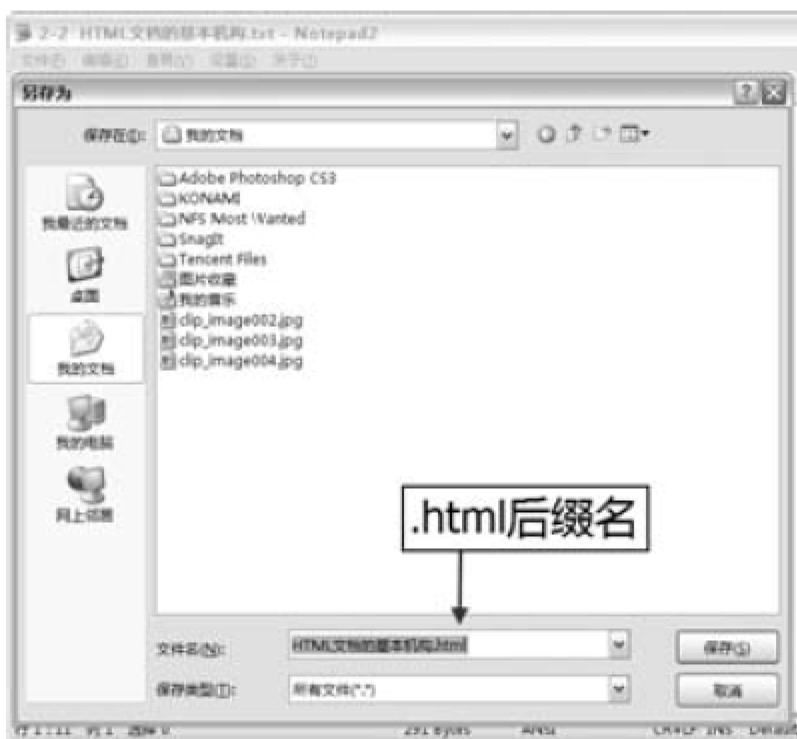


图 1.12　编写一个简单的页面

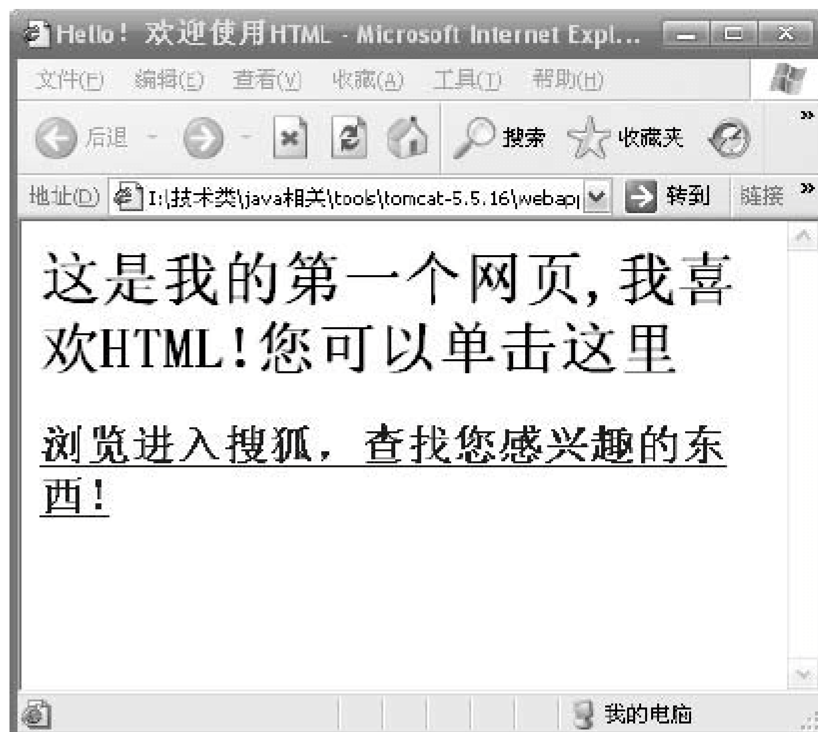


图 1.13　第一个页面效果

【深入学习】输入结束后，在记事本中将文件另存为以.html 为后缀名的网页文件，如图 1.12 所示。将页面保存为一个具体的页面文件存放在计算机中，并命名为“编写一个简单页面.html”。



注意：.html 后缀是需要设计者手动去修改的，系统默认情况下是文本后缀.txt。

最后在浏览器中打开的效果如图 1.13 所示，这就是已经完成的页面。

单击“浏览进入搜狐，查找您感兴趣的东西！”这一链接语句，如果用户的计算机已经连接上互联网，则页面会转入搜狐的主页。

1.6　小　　结

本章是 HTML 的入门部分，主要的知识点有：

- ❑ 网页的概念、静态网页、动态网页及常用的网页开发工具，读者通过这部分可以了解到 HTML 页面的内容形式和开发工具。
- ❑ 通过了解网页浏览器的工作原理和常见的浏览器，可以认识浏览网页的工作方式。
- ❑ HTML、XML 和 XHTML 之间的联系，XML 是语言规范更严谨的 HTML，XHTML 是 HTML 到 XML 的过渡。
- ❑ 最后通过一个 HTML 页面的例子，展示了一个页面从创建到最终成型显示在浏览器中的过程。

在接下来的章节中将为读者揭开 HTML 的神秘面纱，学习 HTML 语言的使用技术。

第 2 章　通过学习他人的网页了解 HTML 能做什么

HTML 最初创建的目的是创建一种文本描述语言，发展至今，成为了一种标记语言，以十分直观的方式告诉浏览器页面中的内容。而在 HTML 出现以前，创建一个可以展示文档内容、包含多媒体信息、具备丰富多彩动态效果的文件是一件难以想象的事情，这意味着如果想通过网络传递信息，只能通过 Word 这类软件。HTML 的出现令很多不擅长使用软件的人一样可以享受网络带来的快捷性和便利性。本章将介绍 HTML 语言的入门基础，主要的知识点如下。

- ❑ 通过怎样的方法来查看网页的源码。
- ❑ HTML 语言的使用方法。
- ❑ 学习 HTML 页面基本的结构和标签。
- ❑ 通过 HTML 页面的实例了解制作简单的页面。

2.1　用记事本打开一个页面

文件扩展名是操作系统用来标记文件格式的一种机制。扩展名一般跟在文件名的后面，由一个分割符号隔开，如 film.avi，.avi 说明 film 是一个视频，扩展名就如同文件的身份说明，区别了文件的类别和作用。

HTML 页面的文件后缀名是.html 或.htm，一般情况下，系统默认使用网页浏览器打开。这种情况下，使用者是无法看到页面代码的，实际上看到的是一个页面制作的最后展示结果。

【本节示例参考：资料光盘\第 2 章\2-1　学习第一个页面.html】

【实例 2-1】通过此例学习第一个页面，其代码展示如下：

程序 2.1　学习第一个页面.html

```
01   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03   <html xmlns="http://www.w3.org/1999/xhtml">
04   <head>
05       <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
06       <title>Untitled Document</title>
07     <style type="text/css">
08     <!--
09       body {
```

```
10          background-image:url(图片/MIL.JPG);              //放入一张背景图像
11          background-repeat: no-repeat;
12               }
13      #Layer1 {
14                position:absolute;                          //确定页面层的位置
15                margin:0;
16                left:247px;
17                top:395px;
18                width:165px;                                //设置页面的宽度
19                height:89px;                                //设置页面的高度
20                z-index:1;
21                }
22           .style1 {
23                  font-size: 24px;                          //定义文本字体的样式
24                  font-weight: bold;
25                  color: #FFFFFF;                           //设置页面文本的颜色
26                  }
27           #Layer2 {
28                  position:absolute;
29                  left:408px;                               //设置 Layer2 在页面中的位置
30                  top:58px;
31                  width:190px;
32                  height:170px;
33                  z-index:2;                                //z 轴方向的高度
34                  }
35           -->
36           </style>
37         </head>
38            <body>
39              <div class="style1" id="Layer1">
40               <form id="form1" name="form1" method="post" action="">
41                           <!—这里是页面中的表单 -->
42                  <label>
43                      <span class="style1">写点什么吧
44                      </span>
45                   <textarea name="textarea" rows="3"></textarea>
46                 </label>
47               </form>
48             </div>
49            <div class="style1" id="Layer2"> 嗯...我想，您会喜欢上做网页的</div>
50          </body>
51         </html>
```

【运行程序】浏览该页面，如图 2.1 所示，这是一个静态页面。



图 2.1　学习第一个页面

【深入学习】通过记事本打开网页文件可以看到代码，右击网页空白处，在弹出的快捷菜单中选择“查看源文件”命令，系统会默认在记事本中打开源码，如程序 2.1 所示，在这个页面的代码中包含了文本内容、CSS 的样式表、一个简单的提交表单。

一个完整的 HTML 文档包含两类事物：标签和页面内容。放在“< >”内即是 HTML 语言标签，用来编辑修饰页面内容。页面内容即指页面开发者想放入页面的文本、图像、多媒体文件等，如程序 2.1 中，页面内容是“写点什么吧”和“嗯...我想，您会喜欢上做网页的”两句话和作为页面背景的图片。

这听起来很可怕，似乎编辑页面需要使用很多标签，使用 HTML 是一件很复杂、很多工序、很麻烦的一件事。事实上，真相远远没有那么复杂。虽然现在还不必了解这段代码的含义，但还是做个简单的说明。

程序 2.1 中，前 3 行代码即<head>之前的部分是代码的声明部分，从第 4～37 行即<head>标签内的代码部分，针对所要设计的页面，<head>标签中定义了两个层的区域：Layer1 和 Layer2，分别是第 13～21 行和第 27～34 行。同时，定义了一个 CSS 样式表，从第 22～26 行，用来编辑文本样式。

从第 38～50 行，即<body>标签内的代码，为页面的主要内容，通俗地说，可以认为<body>内是设计者放入需要通过浏览器展示在互联网上的内容。HTML 并没有很慎密的格式规范和很复杂的算法研究，本书会在后面的章节中继续详细分析程序 2.1。

2.2　初识 HTML

HTML 不是一门程序语言，学习 HTML 不需要任何编程基础，相对于计算机语言如 C++或 Java 来说，要简单得多，当然简单并不意味着不重要，试想，如果互联网没有了网页，该成什么样？简直无法想象。

2.2.1　HTML 语法

程序 2.1 中，有一些单词放在<...>中，看上去很像某种约束好的特殊定式，如果尝试多打开一些页面，会发现很多页面中都有<body></body><title></title>这样的标签。在 HTML 语言中，如定式

<body>…</body>被称为标签，其作用是为了“标记”页面中的内容，使浏览器能够识别设计者的要求，正确地在网页中显示出来。使用标签需要遵循 3 点规则。

- 标签由开始标签起头，一定要有对应的结束标签来收尾，如以<body>起头的一个标签，一定要有</body>来收尾。



注意：有部分标签默认情况下可以省略收尾标签，如<p>、<div>等，但并不表示这里没有收尾标签，只有<link>和<base>两个标签除外，可参考 2.3.3 小节。

- 标签中的属性值必须放在引号内。属性是用来描述一个事物特征的表述语言。例如，这个人的身高竟有 2.26 米。“身高”就是描述这个人的一个属性。“2.26 米”就是这个属性的属性值。

如程序 2.1 中的第 45 行：<textarea name="textarea" rows="3">，表明在这个命名为 textarea 的文本区域行数是 3，3 就是一个属性值。像这种表明属性的值需要放在引号内。

- 对同一文本使用多个标签时必须按照嵌套的原则，即一个标签必须是嵌套在另一个标签内使用。

这就如同写文章时的规则要求，使用大小括号的原则：“{大括号[中括号（小括号）]}”，就是一个典型的嵌套。

2.2.2 HTML 文档的结构

这里用一个类比来描述 HTML 文档的结构，制作一个页面，可以把页面看成是设计者正在描绘的一幅人体的“半身像”。首先，需要“一块画布”用来作画，在文档中，用<html>标签来定义网页的起始和结束。然后，在“画布”内（<html>标签内）编写页面的“头部”（<head>标签部分）和页面的“身体”（<body>标签部分）。这样，一个页面的“形”就建出来了，如图 2.2 所示的对页面结构形象的描述。



说明：具体的 6 个标签将在 2.3 节中详细介绍。

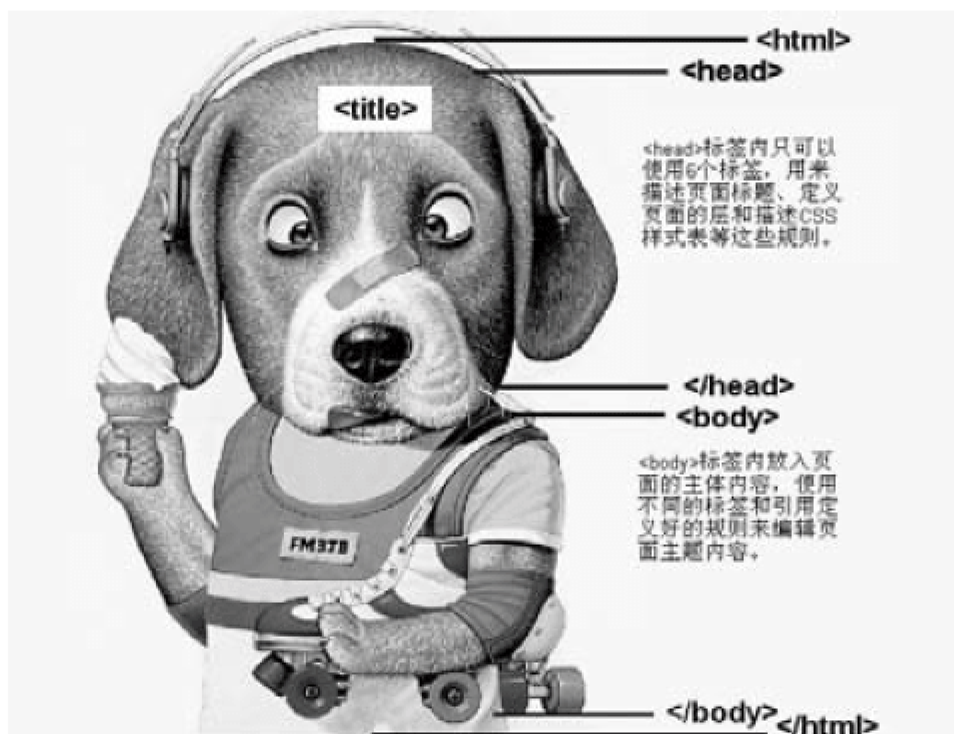


图 2.2 页面代码可分为<head>和<body>

最后，在“勾勒好的形体中”，设计者还需要给这个“形”添加一个“title”，给页面起名。那么

一个 HTML 文档的基本结构在记事本中写出来，就如同程序 2.2 所示了。

【本节示例参考：资料光盘\第 2 章\2-2　HTML 文档的基本结构.html】

【实例 2-2】一个 HTML 文档的基本结构源码的展示如下：

程序 2.2　HTML文档的基本结构.html

```
01   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03   <html xmlns="http://www.w3.org/1999/xhtml">
04     <head>
05      <title>Untitled Document</title>     //head 部分是页面头部
06     </head>
07     <body>                                //body 部分是页面身体
08      </body>
09     </html>
```



注意：<!--　…　-->是在 HTML 文档中对代码的注释，不起任何作用，可以在任何位置放入。



说明：这是一个空白页面，在浏览器中不会展示任何内容。

【深入学习】在这样的结构中，首先遇到文档中的样本代码，如程序 2.2 中的第 1、2 行。然后是文档中的<html>标签，第 3 行和第 9 行。再之后是页面的“头”——<head>标签和“身体”——<body>标签，如代码中第 4～6 行是<head>标签部分，第 7、8 行是<body>标签部分。

这两个标签和<html>标签是“父子”关系，即<html>标签是父级，这两个标签是子级，它们可以放在<html>标签内使用；反之，如果<html>标签放在它们内使用，则不行。

```
<html>
  <head>
  </head>
<html>
```

这样的写法是正确的，但是如果出现下面这种写法：

```
<head>
  <html>
  </html>
</head>
```

这种写法是不可以的。

程序 2.2 中第 5 行的<title>标签就是用来为页面定义标题的，属于<head>标签的下一级，<title>标签通常放在<head>标签内使用，反之则不行，正确的常用形式如下：

```
<head>
  <title>
  </title>
</head>
```

<title>标签就类似于“页面头部内的眼睛”，属于“头”的一部分，<title>标签是<head>标签的子级。所以，一个网页基本结构标签的嵌套关系有 4 层，如图 2.3 所示为页面层级结构。

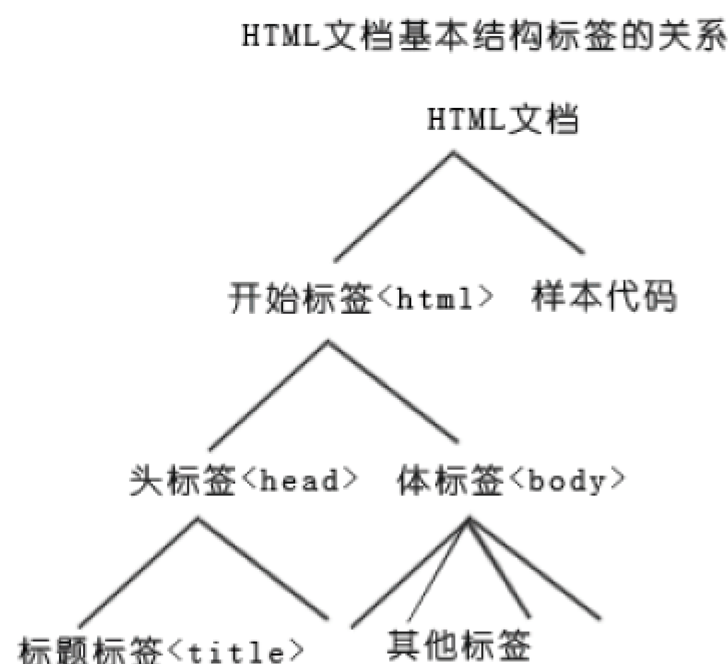


图 2.3　HTML 文档基本结构标签的关系



说明：在<head>和<body>下有许多不同的父子关系的标签，在这里并没有罗列。

当然，如果设计者希望页面被描绘得惟妙惟肖、栩栩如生，就需要了解更多的 HTML 语言作用和其使用方法，但是，任何页面的结构，或者说“半身像”总是由“头”和“身体”组成的，这个结构是不会变的。

2.3　HTML 文档基本结构标签的作用

页面文档的基本结构可分为 4 层关系，这其中涉及 5 个重要的结构性标签用来构成一个页面，分别是样本代码、开始标签、头标签、标题标签和文体标签，一个完整的页面通常必须具备这 5 个标签。

- 样本代码：用来声明代码的版本。
- 开始标签：定义页面从哪里开始到哪里结束。
- 头标签和头标签的对象：有 6 个特殊的标签可以放在头标签中使用。
- 标题标签：设置网页的标题名。
- 文体标签：用来表现网页的主体内容。

2.3.1　给页面一个声明——样本代码

在网页代码文档中，通常最先看到的就是样本代码。如实例 2-3 中的语句，就是一段样本代码的声明。

【实例 2-3】样本代码的展示，其源码展示如下：

```
01    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

【深入学习】样本代码看起来很复杂，似乎很难理解。实际上，它只是起到了一个声明的作用，告诉浏览器，所书写的 HTML 代码的版本。代码中 DOCTYPE 即 Document Type 的简写，意思是文档类型。整段定义了设计者使用文档类型的定义，然后把这些信息传递给浏览器，浏览器根据设计者的定义来解析代码，展现出相应的页面。

这一段的含义是定义了网页文档 XHTML 1.0 Transitional，允许使用 HTML 4 的标签，符合 W3C（万维网联盟）的标准。还有另一种常用的样本代码，声明的网页文档是 XHTML 1.0 Strict。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/ DTD/xhtml1-strict.dtd">
```

XHTML 1.0 Transitional 文档和 XHTML 1.0 Strict 文档的区别在于，如同它们的命名一样，前者是“过渡性”的，允许使用 HTML 4 的标签，而后者丢弃了很多旧的标签。从长远来看，后者是针对于 CSS 的使用原则，严格规定的文档类型。当读者习惯于使用 CSS 来表现网页时，完全可以选择后者。对于初学者来说，为了更容易地理解 HTML 文档，适合于使用 XHTML 1.0 Transitional 文档。



说明：本书中不加特殊说明，都遵循于 XHTML 1.0 Transitional 文档。

此外，还有一种比较常见的声明方式，使用 XHTML 1.1 文档。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1 //EN"
```

这是最新版本的 XHTML 1.1，有时会出现一些浏览器不能识别部分 HTML 标签的问题。样本代码的目的是为了验证代码文档的准确性，只需要在开头声明即可，之后的代码中无须再次声明。

2.3.2　踏出制作页面的第一步——开始标签<html>

从<html>标签开始，页面就进入设计者操作的领域当中，<html>标签的作用就相当于设计者在告诉浏览器，整个网页是从这里开始的，然后到</html>标签这里结束，好像在记事本这个“画板”中来确定设计者的“画布”是哪一个范围。一般来说，<html>标签放在样本标签之后。在严格的网页代码中，通常以如下的形式出现：

```
<html xmlns=http://www.w3.org/1999/xhtml    xml:lang="en">
```



说明：xmlns 和 xml:lang 是 XHTML 网页的标准要求，用于指定与该标签相关的一些信息。

2.3.3　页面的脑袋——头标签和头标签的对象

在 2.2.2 小节中一个页面的结构被类比成一个“半身像”，<head>标签是页面的“头部”。那么从

制作页面的功能上来说，头标签内的对象就如同是“调色板中的颜色”。

<head>标签的对象有些特别，一般来说，只有 6 个标签能放在<head>标签内，除了本章前面看到的标签<title>和<meta>，还有<link>、<base>、<style>和<script>标签，设计者使用这些标签，定义出不同的“颜色”来描绘页面。



注意：<head>标签没有功能性作用，重点是了解<head>标签的对象，即可以放在<head>标签内的是哪些标签，有什么具体的作用。

1. <meta>标签

<meta>是 HTML 文档<head>标签内的一个辅助性标签，<meta>标签有两个重要的属性：name 和 http-equiv，通常用于能够优化页面被搜索的可能。具体的写法格式一般是，在 name 后输入属性，content 后输入对于属性具体描述的词汇。如程序 2.3 所示的<meta>标签的使用。

【实例 2-4】<meta>标签下 name 属性的使用，其源码展示如下：

程序 2.3　页面头部分

```
01  <head>
02          <meta name="keywords" content="nine，twenty-three">
03              <!—搜索引擎的关键字说明 -->
04          < meta name="description"content="……">
05              <!—向搜索引擎描述主要内容-->
06          < meta name="generator"content="Dreamweaver">
07              <!-- 向页面描述生成的软件名 -->
08          < meta name="author" content="depp">
09              <!—向页面说明设计者姓名 -->
10          < meta name="robots" content="all">
11              <!—向页面说明限制搜索的方式-->
12        </head>
```

【深入学习】<meta>标签下不同的属性分别有着不同的作用，它们令页面生动，增强亲和力。

- keywords：向搜索引擎说明页面的关键字，content 后输入供搜索的具体关键字。
- description：向搜索引擎描述页面的主要内容。
- generator：向页面描述生成的软件名，content 后面输入具体的软件名称。
- author：网页的设计者，content 后面输入设计者的具体姓名。
- robots：限制搜索的方式，content 后面通常可输入 all、none、index、noindex、follow、nofollow 这些其中之一，不同的属性分别有不同的作用，限制页面被搜索的方式。

<meta>标签下另一个属性 http-equiv，其作用是反馈给浏览器一些明确的信息，来帮助浏览器更精确地展示页面。如实例 2.5 是<meta>标签下 http-equiv 属性的使用规则。

【实例 2-5】<meta>标签下 http-equiv 属性的使用，其源码的写法如下：

```
01  <head>
02    <meta http-equiv="content-type" content="text/html; charset=gb2312" />
03  </head>
```

【深入学习】代码第 2 行的作用是告诉浏览器，该页面所使用的字符集是 gb2312，即国际汉字码，如果当使用者浏览有这段代码的页面时，电脑内又没有 gb2312 字符集，浏览器会提示使用者，“需要下载 xx 语支持”。

http-equiv 属性下，还有其他的一些常用的特殊效果，如网页定式跳转效果。程序 2.4 表现的是页面自动跳转的功能。

【本节示例参考：资料光盘\第 2 章\2-6　页面自动跳转实例.html】

【实例 2-6】页面自动跳转的实例，其源码展示如下：

程序 2.4　页面自动跳转实例.html

```
01  <html>
02    <head>
03      <title>跳转页面的实例</title>
04      <meta http-equiv="refresh" content="6; url=http://www.baidu.com/">
05      </meta>
06    </head>
07    <body>这个页面在 6s 后自动跳转到百度，1、2、3、4、5、6 ......
08    </body>
09  </html>
```

【深入学习】本例第 4 行是实现页面跳转的代码行。

- refresh：是对属性的具体描述，说明是令页面自动跳转的效果。
- content：后面输入等待的时间，url 后面输入跳转的页面链接地址。
- content="6; url=… "：作用是令页面在 6 秒后页面自动跳转到设定好的某个页面。如果去掉 url 跳转的链接地址，所起的作用令页面每 6 秒刷新一次。



说明：<meta>标签对于一般页面制作者来说，实用意义不大。在大多数网页中，<meta>标签也只是用来定义版权信息。

2．<link>标签

<link>标签定义了一个外部文件的链接，经常被用于链接外部 CSS 样式。如下列语句表明了引用 temp.css 文件的外部链接。

```
<link rel="stylesheet" type="text/css" title="temp" href="/temp.css/">
```

3．<base>标签

<base>标签为整个页面定义了所有链接的基础定位。其主要的作用是为了确保文档中所有的相对 URL，都可以被分解成正确的文档地址，使在文档本身被移动或重命名的情况下也可以正确解析。

```
<base href="http://www.ou-uo.com "/>
```

在创建文档集合时经常使用<base>标签，为了不会破坏文档中的任何链接，使用者通过在每个文档中放置正确的<base>标签，就可以在目录甚至服务器之间移动整个文档集合。



注意：<link>和<base>这两个标签在写法上不需要封闭。

4．<style>标签

<style>标签用来定义 CSS 的样式。使用方法如下列代码，它表明了在页面中引入一个.style 的样式表。

```
<style type="text/css">
<!--
.style1 {
     font-size: 24px;
     font-weight: bold;
     color: #FFFFFF;                  //这里定义了页面文本的字体样式、颜色
         }
-->
</style>
```

<style type="text/css">是 XML 的标准格式，即这样是更严格的写法。一般情况下，省略 text/css 也是可以的，大部分浏览器都能辨识<style>标签，不过并不提倡这种写法，严格的写法才能避免出现疏忽。



说明：样式表是 CSS 学习中一个重要的部分，在本书后面章节有大量的内容介绍 CSS 样式表的运用法则。

5．<script>标签

<script >标签用来定义页面内的脚本，如页面中常用的脚本语言 JavaScript，通过<script>的事件属性来放入 HTML 文档中。使用方法如下：

```
<script type="text/javascript">
</script>
```



说明：<style>和<script>标签会在后面的章节中详细介绍。

2.3.4　给页面起名字——标题标签<title>

<title>也是<head>标签的对象，但如果页面没有标题，那么所有的页面标题都将默认为 Untitled Document。所以，一个正规的页面必须具备页面标题，而<head>标签中的其他 5 个标签可以视情况而选择使用。

<title>标签是本书中的一个表现性标签，什么是表现性标签呢？通俗地说，放在这类标签中的文本，可以通过浏览器展现给浏览者。<title>标签必须嵌套在<head>标签中使用，其作用是用来定义网页的标题，也就是给网页起个名字，如程序 2.5 所表示的<title>标签的使用方法。

【本节示例参考：资料光盘\第 2 章\2-7　<title>标签的使用.html】

【实例 2-7】<title>标签的使用，其源码展示如下：

程序 2.5　<title>标签的使用.html

```
01  <html>
02    <head>
03      <title>title 标签的使用</title>
04    </head>
05  </html>
```

【运行程序】运行该源码，在浏览器中的最终显示效果如图 2.4 所示。

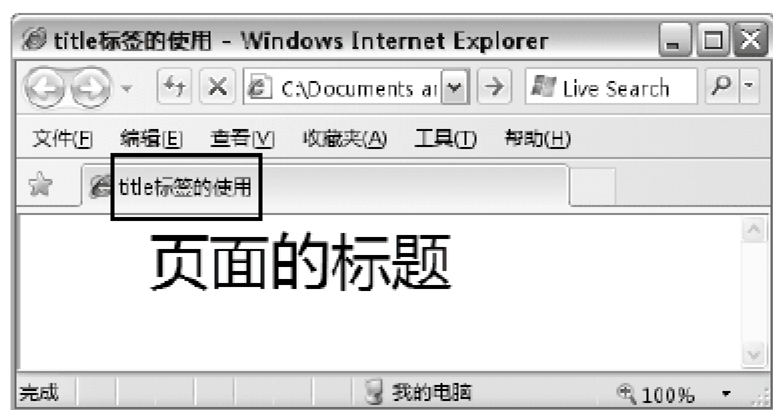


图 2.4　<title>标签的使用

【深入学习】<title>标签可以用来描述这个页面的主要内容，可以使用户方便地添加到收藏夹。给页面起名字是一个好习惯，不仅便于对页面的管理，而且使用户通过页面标题就能大致了解页面的内容。

2.3.5　页面的身体——体标签<body>

如果说<html>标签定义了网页的开始和结束，那么<body>标签的作用是定义了网页主体内容的开始和结束，网页主体内容是指页面浏览者能够在浏览器中看到的网页中的内容。设计者把网页主体内容放入<body>标签内，通过浏览器展示在互联网上。如程序 2.6 中<body>标签内的文本，代码中的第 6 行就是页面的主体内容。

【本节示例参考：资料光盘\第 2 章\2-8　<body>标签的使用.html】

【实例 2-8】<body>标签的使用，其源码展示如下：

程序 2.6　<body>标签的使用.html

```
01  <html>
02    <head>
03      <title>body 标签的使用</title>
04    </head>
05    <body>
06  一天，3 名工程师一同驾车去旅游。半路上车子坏了,抛锚了。机械工程师说: “可能是…
07    </body>
08  </html>
```

【运行程序】浏览该页面，结果如图 2.5 所示。

最终显示在浏览器上的效果如图 2.5 所示，原先代码中的文本出现在页面中。

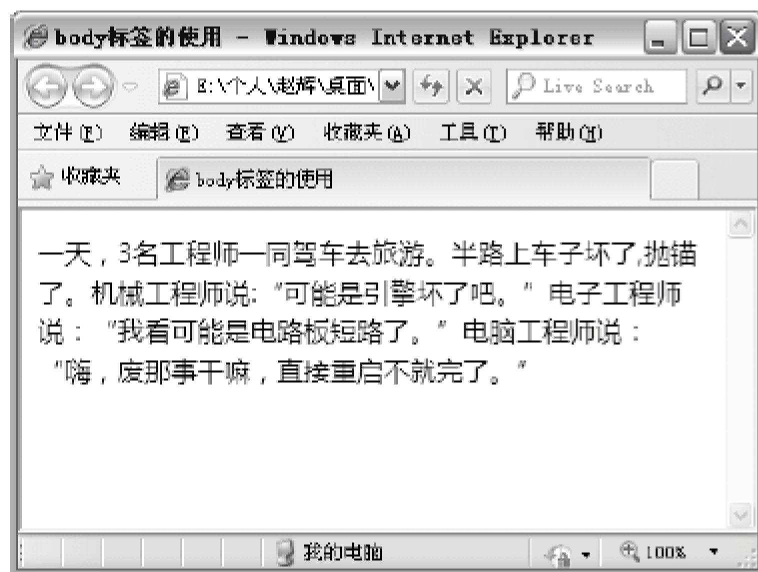


图 2.5　<body>标签的使用

【深入学习】<body>标签中的内容会通过浏览器展示在页面上。但是也要注意，这种简单的文本页面会受到很多的限制性，如文本换行的一些问题。那么如果为了实现文本的换行，在编辑 HTML 实例 2-8 文档时，是不是可以写成这样呢？

```
01  <html>
02    <head>
03      <title>body 标签的使用</title>
04    </head>
05    <body>
06    一天，3 名工程师一同驾车去旅游。半路上车子坏了,抛锚了。
07
08    机械工程师说:“可能是引擎坏了吧。”电子工程师说：“我看可能是电路板短路了。”
09
10    电脑工程师说：“嗨，废那事干嘛，直接重启不就完了。”
11    </body>
12  </html>
```

那么实际情况会不会像代码文档写的那样，文本自动隔开段落呢？实际上，最后的结果还是图 2.5 的样子，浏览器并不会自动识别文本的版式。所以，在<body>标签中，排版文本是不可能的，因此，为了使设计者能对文本做大量的排版修改，需要使用能够适用于<body>内的标签。

<body>标签内的对象有很多，而它们好比是设计者手中的“画笔”，用不同的“画笔”才能描绘出不同形式的图案。



注意：同<head>一样，<body>是用来表明页面的结构，重点是<body>标签的对象，即哪些标签可以放在其中使用和如何使用（不包括<head>内的 6 个标签）。可以说，本书之后的大部分章节都在介绍如何使用<body>标签的对象。

2.3.6　美化 HTML 文档

养成编写格式清晰的 HTML 文档的习惯是很重要的，不仅方便给其他的使用者浏览，也便于自己今后的再修改。好的文档能体现出一个设计者的思维方式。一个好的 HTML 文档应具备以下 3 个方面的条件。

- ❑ 代码使用准确规范，不应有错误的拼写。
- ❑ 代码结构清晰，使人一目了然。
- ❑ 没有错误或者多余不必要的代码出现。

当然，有时代码错误、结构混乱，浏览器一样可以辨识。例如程序 2.6，如果省略<head>标签，把<title>标签放在最后使用，写成如下样式：

```
01	<html>
02	<body>
03	一天，3 名工程师一同驾车去旅游。半路上车子坏了,抛锚了。机械工程师说:“可能是引…
04	</body>
05	<title>body 标签的使用</title>
06	</html>
```

程序 2.6 最终依然能正确地显示在浏览器中，但是，对于设计者来说，这样的习惯很不好，随意的发挥看似提高了编写的效率。实际上，面对大型网站页面编辑，多人协作情况下，代码的混乱会造成很多不必要的错误和返工。所以，从简单的页面开始，养成良好的编写代码的习惯也是成为一个优秀的页面开发人员的必要条件。

2.4　案例：我们的第一个页面

本节的内容是制作一则寓言小故事。首先，定义我们需要制作一个怎样的网页。一个简单的页面，里面有一段文字，这个页面需要标题，设计好之后，就有了一个明确的构思。

（1）找到需要的素材，一个小故事，内容如下：

一只小猪、一只绵羊和一头乳牛，被关在同一个畜栏里。有一次，牧人捉住小猪，它大声号叫，猛烈地抗拒。绵羊和乳牛讨厌它的号叫，便说：“他常常捉我们，我们并不大呼小叫。”小猪听了回答道：“捉你们和捉我完全是两回事，他捉你们，只是要你们的毛和乳汁，但是捉住我，却是要我的命呢！”

立场不同、所处环境不同的人，很难了解对方的感受；因此对别人的失意、挫折、伤痛，不宜幸灾乐祸，而应要有关怀、了解的心情。要有宽容的心！

（2）给网页起一个标题，根据这个小故事的内容，把标题起为《宽容》。

（3）编写 HTML 代码。网页的标题是：

```
<title>宽容</title>
```

然后，网页的主体内容就是这个小故事，应该放在<body>标签内。

```
<body>“一只小猪、一只绵羊和一头乳牛，被关在同一个畜栏里。有一次，牧人捉住小猪，它大声号叫，
… …
但是捉住我，却是要我的命呢!<br>
立场不同、所处环境不同的人，很难了解对方的感受，因此对别人的失意、挫折、伤痛，不宜幸灾乐祸，而应要有关怀、了解的心情。要有宽容的心！”
</body>
```



**说明：文本中出现了
标签，它的作用是令文本换行，本书第 3 章中会详细介绍文本排版的使用。**

（4）需要加入样本代码，一些必需的结构化标签，最后搭在一起的完整代码如程序 2.7 所示。

【本节示例参考：资料光盘\第 2 章\2-9　一则寓言故事的页面.html】

【实例 2-9】设计一则寓言故事的页面，其源码展示如下：

程序 2.7　一则寓言故事的页面.html

```
01  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03  <html xmlns="http://www.w3.org/1999/xhtml">
04    <head>
05      <title>宽容</title>
06    </head>
07    <body>
08  一只小猪、一只绵羊和一头乳牛，被关在同一个畜栏里。有一次，牧人捉住小猪……
09    </body>
10  </html>
```

【运行程序】这样，整个页面的编辑通过 4 步流程完成，最终显示在浏览器中的结果如图 2.6 所示。

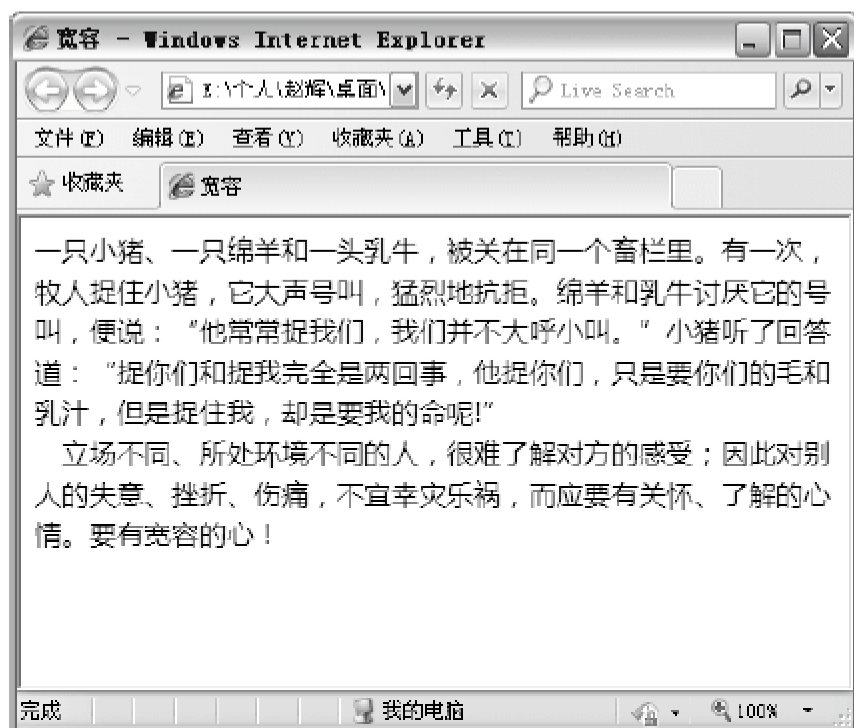


图 2.6　一则寓言故事

【深入学习】通过这个页面介绍了一个简单的设计如何从零最后成为一个页面的过程，设计的想法是基于对页面制作技艺的掌握。也许将这个页面整体拆分开，每一步都简单易于掌握，但是如果想做到将这些知识捏合在一起付诸于实践，这是需要不断地练习来实现的。

2.5　小　　结

本章介绍了开发页面时需要具备的知识点，通过本章的学习，可以对 HTML 语言有个整体的了解，

可以由理解知识走向运用知识的第一步，可以亲手去制作一个属于自己的简单页面。本章的主要知识点有：

- 页面代码编辑器——记事本。通过记事本可以浏览页面代码，开发网页。
- 了解 HTML 页面中使用标签的 4 个基本的规则。
- 了解一个 HTML 页面基本结构和基本结构的标签。这里涉及一个重要的概念：HTML 页面主要由<head>标签部分和<body>标签部分组成，而<head>标签的对象目的是为了表现页面，<body>标签的对象目的是为了展示页面的内容。其关系如同“调色板”和“画笔”。
- 最后通过一个简单的实例，介绍了一个简单页面从构思到做出设计，从设计到给出方案，从实施方案到最后成品完成，最后在浏览器中查看结果。

通过本章的学习，读者已经理解如何把页面内容放入到页面中去，在接下来的章节中，学习如何在页面中编辑、修饰放入的文本。

第 3 章　动手在网页中写些什么

静态页面中的绝大部分内容由 4 类属性的物质组成：文本、图像、视频音频等多媒体文件和超链接。本章从编辑文本的知识点开始介绍。那什么是文本呢，从词源上说，表示“编织的东西”。当然这里说的文本既不是指纹理编织的页面，也不是指某种计算机语言，很简单，是指“书写下来的任何语言”。更通俗地说，本章将学习如何在页面中编写设计者想说的话。本章的知识点如下。

- 了解 HTML 语言：区分清楚旧的使用规则和新规则 CSS 之间的联系和不同。
- 文本排版格式：学会如何使用标签实现在页面中规范写作格式。
- 文本样式：学会如何改变页面中的文本的基本属性和如何使用一些特殊的符号。
- 文本列表：学会在页面中使用无序列表和有序列表来罗列条目。
- 通过实例，学会在页面中编写文本、项目列表。

3.1　新旧方法对比

早期的 HTML 版本中使用一些特殊的标签来编辑文本，如使用...标签来强调突出文本，使文本具有斜体字的效果。但是这种方法使用起来非常繁琐，对于不同段落的编辑需要重复使用，工作量很大，也给之后的再次修改带来很多麻烦。随着 HTML 的发展，设计者发掘了更多的标签来美化文本。1994 年，一位叫哈坤・利的设计者提出用“层叠式”概念的样式表来编辑文本。在 1996 年底，这种方法被正式推出，发展至今就是现在大家比较熟悉的 CSS。其就像是哈里波特的魔法，令本来需要一步步去完成的工序念动一下咒语，瞬间就改头换面。这种奇妙的技术难道不令人感到兴奋吗？

下面通过一个小例子来感受一下新旧方法的区别。如果读者现在对 CSS 完全不了解、看不懂也没有关系，后面的章节将会详细介绍，这里有个感性的认识即可。在早期的旧方法中，使用大量的特殊标签来编辑文本，如程序 3.1 使用的是旧方法。现在，在新方法中，使用 CSS 来编辑文本，如程序 3.2 所示。

【本节示例参考：资料光盘\第 3 章\3-1　旧方法：使用特殊标签.html】

【实例 3-1】旧方法——使用特殊标签，其源码展示如下：

程序 3.1　使用特殊标签.html

```
01    <html xmlns="http://www.w3.org/1999/xhtml">
02      <head>
03        <title>使用旧方法
04              <! --页面的标题 -->
05        </title>
06      </head>
```

```
07   <! --以上是页面的头部，以下是页面的身体 -->
08        <body>
09
10
11        </body>
12   </html>
```

<body>中使用了大量 HTML 标签



说明：<h2>、<em>、<strong>这 3 个标签依次定义了文本字体大小、斜体、加粗的效果。需要说明的是，这里相同的标签编写了两次。

【本节示例参考：资料光盘\第 3 章\3-2　新方法：使用 CSS.html】

【实例 3-2】新方法——使用 CSS，其源码展示如下：

程序 3.2　使用CSS.html

```
01   <html xmlns="http://www.w3.org/1999/xhtml">
02   <head>
03          <title>使用新方法
04     <! --页面的标题 -->
05          </title>
06   <style type="text/css">
07   <!--
08   .style1 {
09          font-size: xx-large;                //设置字体大小
10          font-style: italic;                 //设置字体样式
11          font-weight: bold;                  //设置字体样式
12   }
13   -->
14     </style>
     </head>
<! --以上是定义页面的头部，用来定义修饰页面的样式表，以下是页面的结构，使用样式表来布局页面 -->
     <body>
                      大家好，html 语言并不难</p>
                      努力一定能学得很好 </p>
     </body>
   </html>
```

<body>中的对象引用使用了 .style1 样式表



说明：style {}是引用 CSS 样式表的一种特定格式，font-size、font-style、font-weight 依次定义了文本字体大小、样式、粗体。class 语句是对 style1 样式的调用。

【运行程序】实例 3-1 和实例 3-2 在浏览器中的查看结果如图 3.1 所示。

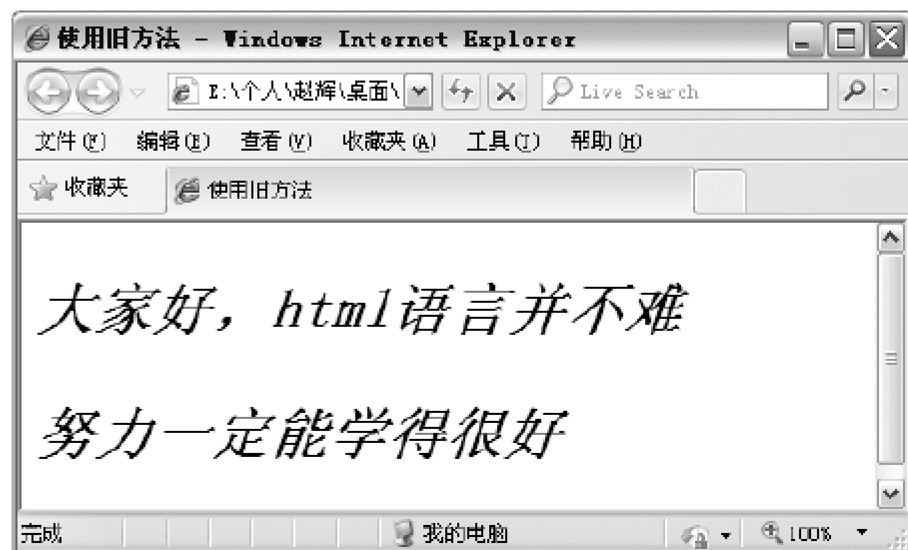


图 3.1　新旧方法在浏览器中查看的结果



注意：程序 3.1 和程序 3.2 在浏览器中查看的结果是一样的。

【深入学习】在实例 3-1 旧方法中，对“大家好，html 语言并不难”编辑时，使用了<h2>、<em>、<strong>标签。对“努力一定能学得很好”这句编辑时，为了实现同上一句同样的效果，再次使用了<h2>、<em>、<strong>标签。注意这里只重复编写了两次同样的标签，编写者不得不重复相同的工作，不仅工作效率低而且代码不便于管理。而 CSS 就像在念动咒语一样，调用了语句 class="style1"，文本就发生了惊人的改变。

正是 CSS 的出现，解决了这种繁琐的重复编写的工作，它彻底将页面的表现和页面的结构两者完全划清界线，虽然这样增加了前端页面开发的难度，但却带来了更快的效率。不仅如此，CSS 还能实现更多的优秀效果。

3.2　文本的排版格式

HTML 文档中使用不同的标签来设计文本的排版，试想如果在阅读一份报纸时，人们可以忍受这份报纸字体大小不一、标题位置混乱、排版更是无比糟糕的效果吗？如图 3.2 这样的排版，甚至找不到它的标题在哪里。

巴黎
只需一件鲜艳单品就能让冬季造型远离沉闷，它可以是一条牛仔裤、一顶贝雷帽，也可以是一头火焰般的长发。

纽约
款式经典甚至老式的针织衫都能带给你一个温暖又时髦的冬季，复古还是前卫则取决于下半身穿什么

图 3.2　糟糕的版式

像样的文本排版需要注意一些重要的细节，如字体大小、行距、段落间距、每行字数等。本章会在后面的小节中穿插介绍这些排版细节，一个工整清洁的文本排版才会令人舒适。

3.2.1　写一行换一行

一般页面文本每行的字数在 35 字左右，这个没有一定的规定，原则是只要每行控制在恰当的长度内就可以了。太短了浏览者需要频繁阅读下一行，太长了浏览者又需要左右移动视线，这样会令浏览者在阅读页面文本时感到不舒服。在 HTML 文档中，可以使用以下的<p>标签或
标签方式使文本换行。它们的写法是：

```
<p>...</p>
<br>...</br>
```

<p>标签对文本的定义是<p>...</p>内的文本是一个段落，如程序 3.3 所示为<p>标签在 HTML 中的使用方法。

【本节示例参考：资料光盘\第 3 章\3-3　使用<p>标签.html】

【实例 3-3】使用<p>标签，其源码展示如下：

程序 3.3　使用<p>标签.html

```
01  <html>
02    <head>
03        <title>如何使文本换行</title>
04    </head>
05          <body>
06          MSN Space 对 HTML 语言具有相当强大的支持能力。<p>虽然处理日志时，MSN
07  Space 只提供给大家简单的几个文字处理功能，但如果你事先在网页编辑器中对文字图片作预先处理，
08  <p>那么通过简单的复制粘贴后，便能让你的日志看上去更生动活泼。
09          </body>
10  </html>
```


标签对文本的定义是
...</br>内的文本从
标签后另起一行，在 HTML 中的使用方法如程序 3.4 所示。

**【本节示例参考：资料光盘\第 3 章\3-4　使用
标签.html】**

【实例 3-4】使用
标签，其源码展示如下：

程序 3.4　使用
标签.html

```
01  <html>
02    <head>
03        <title>如何使文本换行</title>
04    </head>
05          <body>
06          MSN Space 对 HTML 语言具有相当强大的支持能力。<br>虽然处理日志时，MSN
07  Space 只提供给大家简单的几个文字处理功能，但如果你事先在网页编辑器中对文字图片作预先处理，
08  <br>那么通过简单的复制粘贴后，便能让你的日志看上去更生动活泼。
09          </body>
10  </html>
```



说明：<p>...</p>和
...</br>标签默认情况下可以省略</p>和</br>。

【运行程序】浏览这两个页面，实例 3-3 和实例 3-4 在浏览器中的效果如图 3.3 和图 3.4 所示。

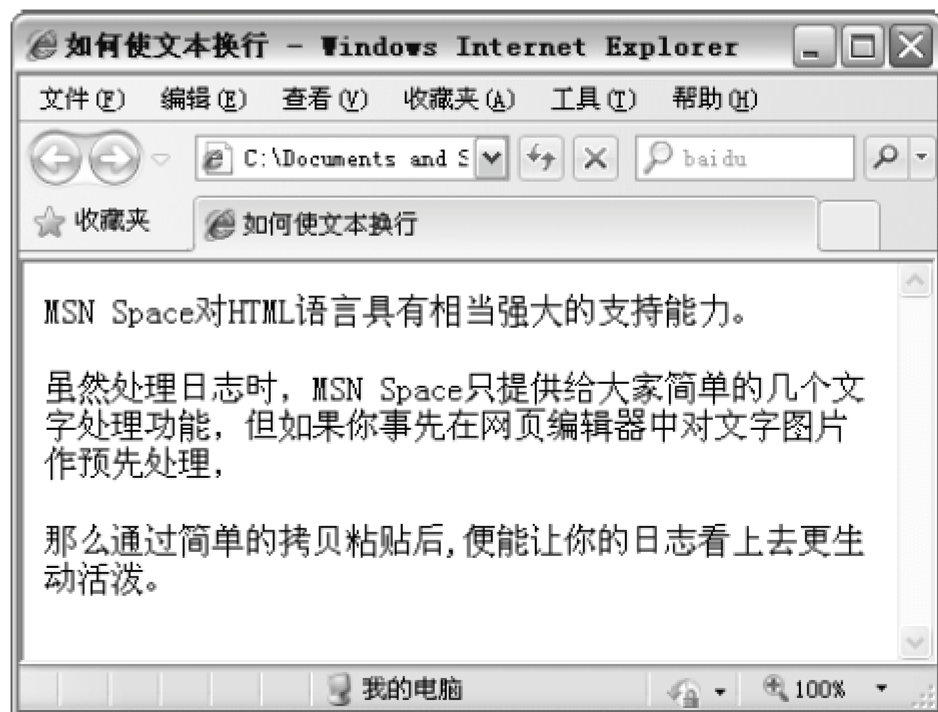


图 3.3　使用<p>标签

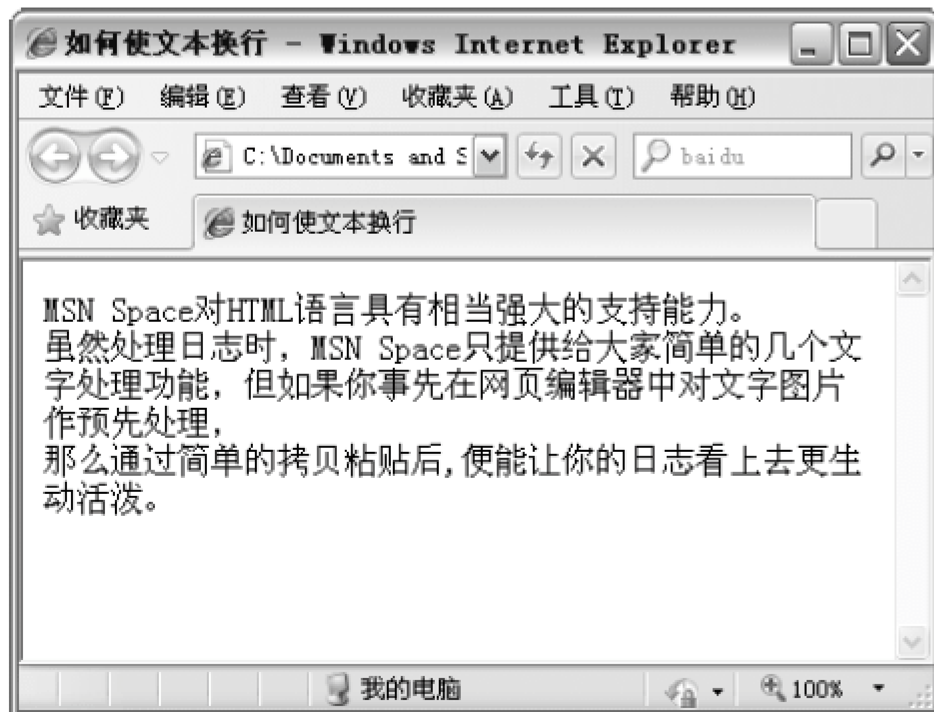


图 3.4　使用
标签

【深入学习】使用<p>标签使文本换行时行距是单倍行距，而使用
标签时，文本换行行距是 0 倍行距。

3.2.2　在页面文本中空格

在正规格式的文本中每一段落的开头会空两格，而在 HTML 源文档中，连续输入的空格键会被默认为只有一个空格，所以，这时就需要使用特殊的空格符号放置在文本中。空格符号的写法是：

```
 
```

使用时在文本中需要输入空格的地方输入“ ”即可，如程序 3.5 所示，在同一句文本中分别放入不同长度的空格数，以此来观察中间的区别。

【本节示例参考：资料光盘\第 3 章\3-5　使用空格符号.html】。

【实例 3-5】在页面中使用空格符号，其源码展示如下：

程序 3.5　使用空格符号.html

```
01   <html>
02     <head>
03         <title>使用空格符号</title>
04     </head>
05     <body>
06         <p><font size="+3">空格 123 符号</font></p>
07         <p><font size="+3">空格     符号</p>
08         <p><font size="+3">空格   符号</p>
09     </body>
10   </html>
```



说明：第 7 行代码“空格　　符号”中，输入了 3 次空格，第 8 行代码“空格   符号”中，输入了 3 次“ ”。

【运行程序】在浏览器中查看的结果如图 3.5 所示。

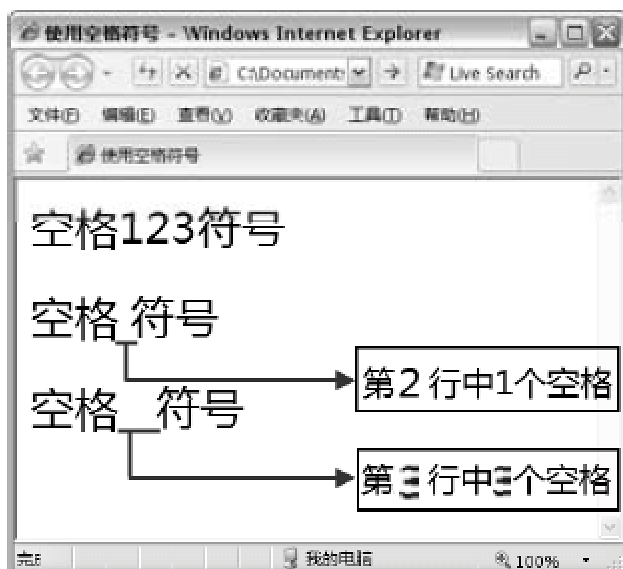


图 3.5　使用空格符号



说明：第 2 行的“空格 符号”中并没有体现出 3 个空格，实际上只是 1 个空格的间距。

【深入学习】空格符号属于 HTML 中的一种特殊符号，如果设计者不想使用“ ”这种特殊符号的写法，同时又希望保留连续的空格符，可以使用一个特殊用途的标签——等宽字体。该标签的写法是：

```
<pre>…</pre>
```

<pre>标签的作用是使文本格式化为等宽字体，不仅能够保留文本中的空格符，也能使文本分行被保留，如程序 3.6 所示就是使用<pre>标签来实现的页面文本排版。在浏览器当中显示的结果如图 3.6 所示。

【本节示例参考：资料光盘\第 3 章\3-6　使用<pre>标签.html】。

【实例 3-6】使用<pre>标签，其源码展示如下：

程序 3.6　使用<pre>标签.html

```
01  <html>
02    <head>
03       <title>使用<pre>标签</title>
04    </head>
05    <body>
06         <pre>
07       君不见黄河之水天上来，奔流到海不复回。
08
09       君不见高堂明镜悲白发，朝如青丝暮成雪。
10
11          人生得意须尽欢，莫使金樽空对月。
```

```
12
13                天生我材必有用，千金散尽还复来。
14            </pre>
15        </body>
16    </html>
```



注意：页面中的文本不会随着浏览器窗口的大小自动换行。

【运行程序】浏览该页面，结果如图 3.6 所示。

【深入学习】如果删除实例 3-6 中的<pre>标签，即第 6 行、第 14 行的代码，最后在浏览器中显示的结果如图 3.7 所示。

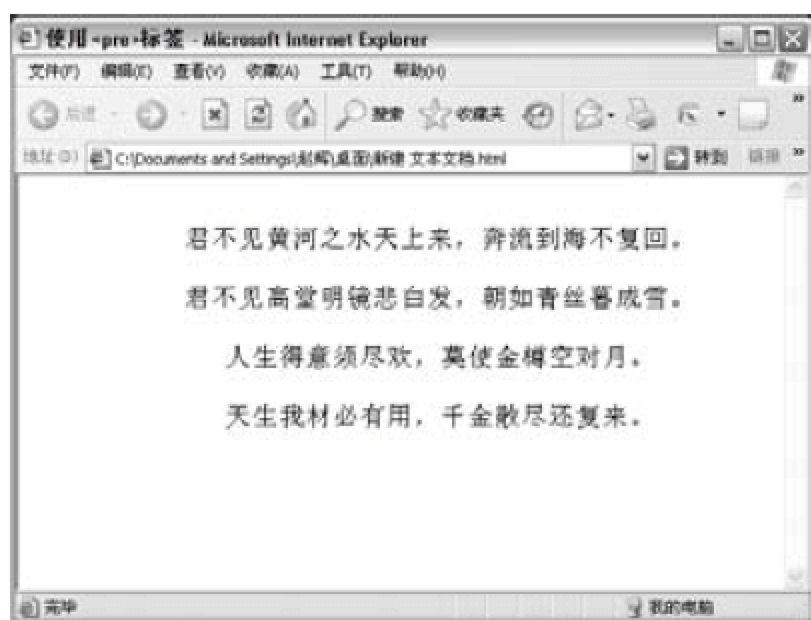


图 3.6　使用<pre>标签

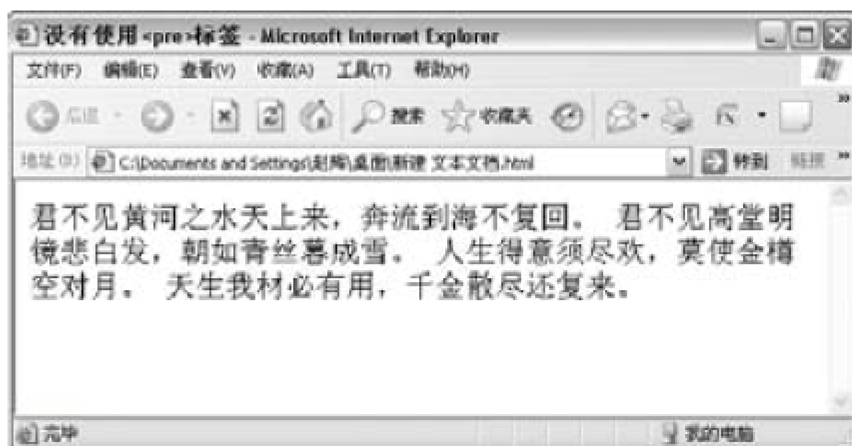


图 3.7　没有使用<pre>标签

通过对比图 3.6 和图 3.7，发现在没有<pre>标签的文本中代码不辨识连续的空格符号，也不能辨识文本的换行。如果设计者是用记事本来编写代码，当然<pre>会显得实用方便一些，但是在大部分可视化页面文档编辑的软件中，如 Dreamweaver，使用这类软件编写时则很少出现<pre>标签。设计者可以根据自己的习惯，结合不同的方法来选择是否要使用<pre>标签。



注意：<pre>标签尽量不要大量地使用，因为这样会给日后修改代码带来不必要的麻烦。

3.2.3　文本的段落要对齐

编排版面时，时常需要使一系列的文本段落按照统一格式对齐，如左对齐、右对齐和居中对齐。实际操作时，设计者当然不是使用一堆堆的空格符号来一格格地编排文本位置。在 HTML 文档中，文本的对齐是通过<align>标签来实现的，通常把 align 放在<p>标签内使用，如下所示：

```
<p align=left>...</p>          <! --左对齐  -->
<p align=center>...</p>        <! --居中对齐  -->
<p align=right>...</p>          <! --右对齐  -->
```

在 HTML 中使用对齐属性的示例如程序 3.7 所示，其将不同文本采用不同的对齐方式放置在页面中。

【本节示例参考：资料光盘\第 3 章\3-7　使文本段落对齐.html】

【实例 3-7】使文本段落对齐的方法，其源码展示如下：

程序 3.7　使文本段落对齐.html

```
01   <html>
02      <head>
03          <title>如何使文本段落对齐</title>
04      </head>
05          <body>
06             <p>文本段落左对齐
07             <p align=left>文本段落左对齐
08             <p align=center>文本段落居中对齐
09             <p align=right>文本段落右对齐
10          </body>
11   </html>
```



说明：默认情况下可以省略</p>。

【运行程序】浏览该页面，结果如图 3.8 所示。



图 3.8　使用段落对齐的效果

【深入学习】实际上，<p>标签在默认情况下相当于<p align=left>，所以在浏览器中查看的结果如图 3.8 所示，默认情况下的文本是和窗口的左边对齐。

如果在编辑文本时对于所有的文本都要求按同一种对齐方式，那么在使用的过程中可以对文本进行全局定义，而并不需要对每段文本添加属性命令。如代码第 7～9 行中是不需要对<p>标签内的每一段文本依次分别定义，而是可以直接放在<body>中使用，如程序 3.8 所示为对整体属性定义和对局部属性定义同时存在的效果。

【本节示例参考：资料光盘\第 3 章\3-8　对整体和局部文本使用对齐命令的效果.html】

【实例 3-8】对整体和局部文本使用对齐命令的效果，其源码展示如下：

程序 3.8　对整体和局部文本使用对齐命令的效果.html

```
01   <html>
02      <head>
```

```
03          <title>标签中对齐属性的应用</title>
04      </head>
05      <body style="text-align:center">
06          <h3>蜀相</h3>
07          <p style="text-align:right">（唐）杜甫                    //使用对齐命令
08          <p>丞相祠堂何处寻，锦官城外柏森森。
09          <p>映阶碧草自春色，隔叶黄鹂空好音。
10          <p>三顾频烦天下计，两朝开济老臣心。
11          <p>出师未捷身先死，长使英雄泪满襟。
12      </body>
13  </html>
```

【运行程序】浏览该页面，结果如图 3.9 所示。

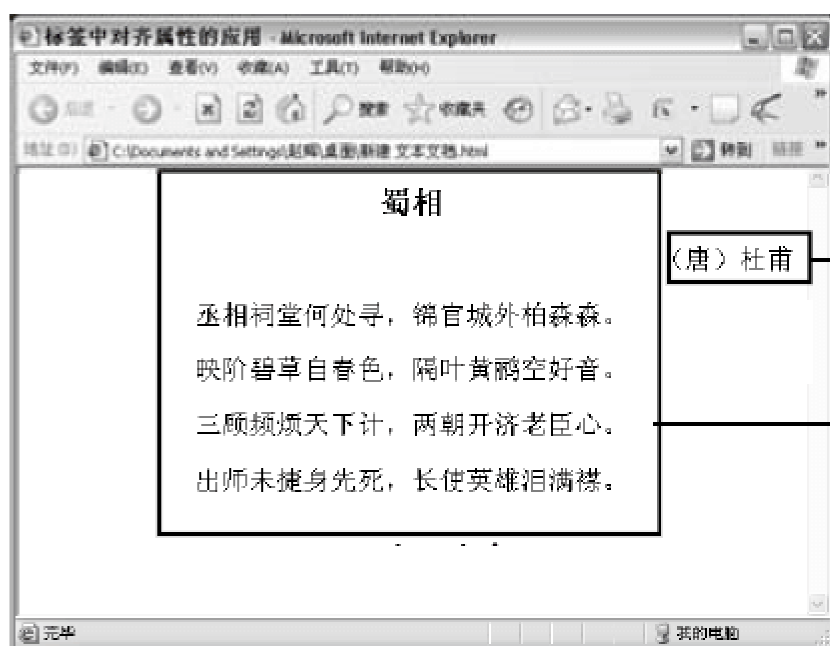


图 3.9　对整体和局部文本使用对齐命令的效果

【深入学习】本例第 5 行对所有<body>标签中的文本定义为居中对齐，这样就不需要依次对每一个<p>标签中的文本进行定义。同时在全局定义下，代码中还有对局部的定义。如代码第 7 行中，文本被定义为右对齐。那么代码第 7 行中的文本最后是居中对齐还是右对齐呢？如图 3.9 所示为浏览器中的结果。

结果是代码第 7 行中的文本在浏览器中是居右对齐的。所以 HTML 语言中有这样的特性：当出现两个或两个以上同属性作用的标签对同一个对象作用时，那么这个对象就依照就近原则，离它最近的那个标签属性起作用。例如，在程序 3.8 中，<body>中添加了居中对齐的属性命令，本来应该对所有文本都起到居中对齐的命令，而代码第 7 行的<p>中添加了右对齐的属性命令，它也对第 7 行的文本起作用。这种情况下，按照就近的原则文本以右对齐的形式表现了出来。

3.3　文本的属性样式

在 HTML 页面中，可以通过使用不同的标签来修改文本的属性，使页面内容起到不同的显示效果。如果说 3.2 节是讨论了如何使文本版块的布局更漂亮，那么本节将学习如何使文本自身打扮得更漂亮丰富些，更好地帮助页面设计者把想说的内容传达给阅读者。

3.3.1 不一样的文本字体大小

早期的 HTML 语言中，有很多不同的标签来实现粗体、斜体以及一些特殊的文字字体。虽然大部分现在已经很少使用，但是一些特殊的标签，如粗体、斜体等，对于理解学习 HTML 语言有很好的帮助作用。下面是本节的知识点，把需要编辑的文本放入相应的标签内，来改变文字效果，如使用<em>标签。

```
<body>
   <p>
       <em>文本内容</em>
 </body>
```

说明：当 CSS 出现后，许多标签渐渐被淘汰，但并非所有标签都抛弃了，一些常用的编辑文本的标签如表 3.1 所示。

表 3.1 改变字体样式的标签

标　签	说　明
<h?>...</h?>	?代表从1～6，依此改变字体从小到大
...	强调文本内容，通常是粗体
...	强调文本内容，通常是斜体
...	粗体字
<i>...</i>	斜体字
<u>...</u>	加下划线
<var>...</var>	变数，通常是斜体字
<cite>...</cite>	引文，通常是斜体字
<dfn>...</dfn>	定义，通常是斜体字，并不是所有浏览器都支持
<address>...</address>	地址，通常是斜体字
<tt>...</tt>	打字机等宽字体
<samp>...</samp>	样本
<code>...</code>	显示原始码使用
<kbd>...</kbd>	键盘输入
<strike>...</strike>	加删除线
<big>...</big>	稍大字体
...	稍小字体
^{...}	数学标记中的上标记
_{...}	数学标记中的下标记

对照表 3.1 的标签排列，如程序 3.9 所示在浏览器中改变字体样式的标签，最终得到的效果如图 3.10 所示。

【本节示例参考：资料光盘\第 3 章\3-9 改变字体样式的标签.html】

【实例 3-9】学习改变字体样式的标签，其源码展示如下：

程序 3.9　改变字体样式的标签.html

```
01  <html>
02    <head>
03      <title>改变字体样式的标签</title>
04    </head>
05    <body>
06        <br><h1>h1 标签内文本样式</h1>
07        <br><strong>strong 标签内文本样式</strong>
08        <br><em>em 标签内文本样式</em>
09        <br><b>b 标签内文本样式</b>
10        <br><i>i 标签内文本样式</i>
11        <br><u>u 标签内文本样式</u>
12        <br><var>var 标签内文本样式</var>
13        <br><cite>cite 标签内文本样式</cite>
14        <br><dfn>dfn 标签内文本样式</dfn>
15        <br><address>address 标签内文本样式</address>
16        <br><tt>tt 标签内文本样式</tt>
17        <br><samp>samp 标签内文本样式</samp>
18        <br><code>code 标签内文本样式</code>
19        <br><kbd>kbd 标签内文本样式</kbd>
20        <br><strike>strike 标签内文本样式</strike>
21        <br><big>big 标签内文本样式</big>
22        <br><small>small 标签内文本样式</small>
23        <br>数字标签<sup>sup</sup>
24        <br>数字标签<sub>sub</sub>
25    </body>
26  </html>
```

【运行程序】浏览该页面，结果如图 3.10 所示。



图 3.10　常用文本标签在浏览器中的查看效果



注意：<strong>和<b>虽然显示的结果相同，但是前者属于逻辑标记，后者属于实体标记。逻辑标记在有些浏览器中不一定能准确显示，而实体标记则显示固定的效果。上述标签中，属于逻辑标记的有<strong>、<em>、<var>、<cite>、<dfn>、<address> 、<code>、<kbd>、<samp>、<tt>标签，属于实体标记的有<i>、<b>、<u>标签。

【深入学习】现在读者应该可以更好地理解本章第 3.1 节程序 3.1 中的例子了。接下来不妨再多思考一个问题，参照程序 3.3 的例子，在 CSS 中是怎样编辑文本的样式呢？<font>标签是不是用来描述文本的属性呢？本书会在之后的 CSS 篇章详细谈到这些问题。

3.3.2　奇妙的特殊符号

在 3.2.2 小节中介绍了“ ”空格符号。事实上，在 HTML 语言中，类似空格符号这样的标记有很多。一般情况下，它们不是经常使用，所以大部分并不常见。但是在某些需要的时候，设计者又不得不使用这些特殊符号，所以了解这些特殊符号并没有坏处。

特殊符号通常有它固定的格式，基本格式为“&...;”。例如，两个重要的很有意思的特殊符号：注册商标“®”和版权商标“©”，如程序 3.10 所示将这两个特殊符号放在了页面中。它们在浏览器中显示的效果如图 3.11 所示。

【本节示例参考：资料光盘\第 3 章\3-10　注册商标®和版权商标©.html】。

【实例 3-10】两个特殊符号在页面中的使用——注册商标®和版权商标©，其源码展示如下：

程序 3.10　注册商标®和版权商标©.html

```
01  <html>
02    <head>
03      <title>注册商标和版权商标</title>
04     </head>
05       <body style="text-align:center">
06             <p>注册商标&reg;
07              <p>版权&copy;
08       </body>
09  </html>
```

【运行程序】浏览该页面，结果如图 3.11 所示。



图 3.11　注册商标和版权商标



说明：表 3.2 给出一些常用的特殊符号，有兴趣的读者不妨自己动手编写一下，看看它们的效果。

表 3.2　常用的特殊符号

字　　符	代　　码
引号	"
和号	&
英镑符号	£
竖直线	¦
节号	§
度数符号	°
加减号	±
上标2	²
上标3	³
乘号	×
除号	÷
AE组合	Æ
Ae组合	æ
二分之一	½
四分之一	¼
居中的点	·

3.3.3　给文本加标注

在书本中，如果需要给一段文章中的某一个名词标加注释，只能在那段文字右上角添加编注，然后在页脚的位置给出解释。而网页的一个优势是设计者可以使用很多奇特的标签，来做到一些现实中书本无法做到的效果。如在网页中，如果设计者希望对某一个名词或某一段文字添加注释，可以用<acronym>标签，使用形式如下：

```
<acronym title="…">…<acronym>,
```

注释的内容放在 title 属性后的引号中，被注释的内容放在标签内。如程序 3.11 所示为给一段文本添加标注，藉此来帮助阅读页面的浏览者做更详细的介绍。

【本节示例参考：资料光盘\第 3 章\3-11　使用<acronym>添加标注.html】。

【实例 3-11】使用<acronym>添加标注的方法，其源码展示如下：

程序 3.11　使用<acronym>添加标注.html

```
01   <html>
02      <head>
03         <title>用<acronym>标签添加标注</title>
04      </head>
05      <body>
```

```
06   自 1851 年英国伦敦举办第一届展览会以来，
07         <acronym title="世博会全称为世界博览会，世界博览会是由一个国家的政府主办，…
08         </big></acronym>
09   因其发展迅速而享有“经济、科技、文化领域内的奥林匹克盛会”的美誉。按照国际展…
10      </body>
11   </html>
```

【运行程序】在浏览器中显示的效果如图 3.12 所示。

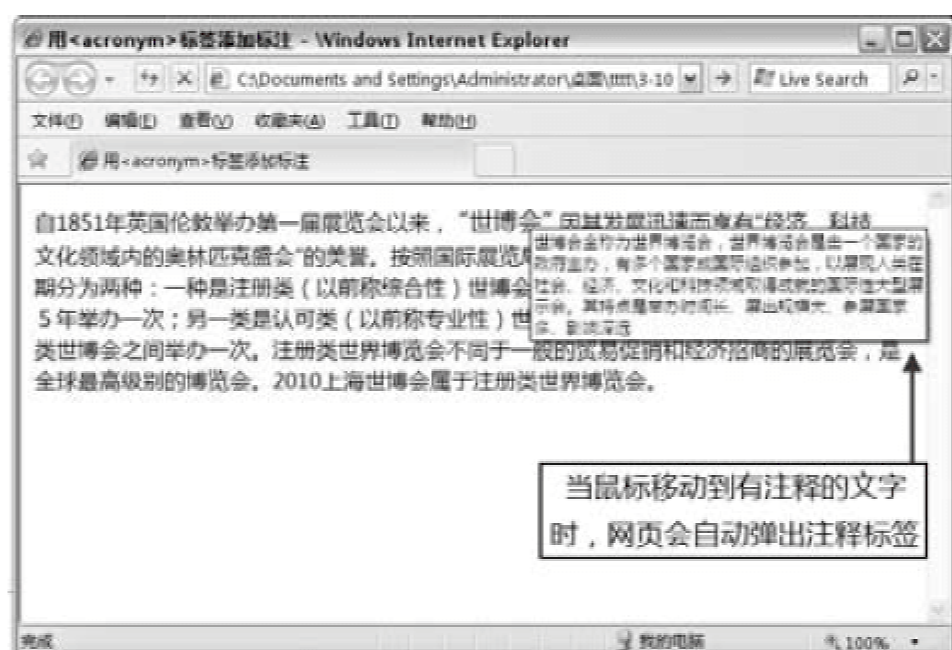


图 3.12　使用<acronym>给文本加注释

【深入学习】本例第 7～9 行中，<big>标签用来编辑“世博会”的字体大小，使字体相对于默认字体稍稍显得大一些。在页面中添加注释时，把被注释的对象放在<acronym>标签内，如代码中的“世博会”这个名词，而对它解释的内容则放在<acronym>内 title 属性的后面。

3.4　整齐的文本列表

就像一本书一定要有目录一样。网页上的信息时常也需要用列表的形式表现出来，如一些目录列表、菜单介绍、计划类条目、罗列并列关系的段落，以此来帮助浏览者更便捷地获取信息，这就是文本列表。页面中文本列表可以分为无序列表、有序列表和定义列表。合理使用列表，不仅能传达页面的信息，有时还能起到美化网页的作用。

3.4.1　无序列表

没有编号的列表称为无序列表，无序列表经常见于项目说明，这是一种并列关系的列表。如果结合 CSS 的修饰作用，它还可以表现为导航栏，在页面中的作用可以说是相当有意义。无序列表以
标签开始，至
标签结束，在<ul>标签中，还需要使用<li>标签来定义列表的每一行，具体的写法如下所示：

```
<ul>
  <li>……</li>
  <li>……</li>
  <li>……</li>
</ul>
```

使用这样的列表可以实现做一些有趣的简介类条目，如程序 3.12 所示就表现了这样一个信息登录的页面。

【本节示例参考：资料光盘\第 3 章\3-12　制作无序列表.html】。

【实例 3-12】制作无序列表的方法，其源码展示如下：

程序 3.12　制作无序列表.html

```
01   <html>
02     <head>
03       <title>制作无序列表</title>
04     </head>
05     <body style="text-align:center">
06       <h3>个人简介</h3>
07       <ul>
08         <li>姓名：_________
09         <p>
10         <li>年龄：_________
11         <p>
12         <li>性别：_________
13         <p>
14         <li>爱好：_________
15         </ul>
16     </body>
17   </html>
```

【运行程序】浏览该页面，结果如图 3.13 所示。



图 3.13　无序列表

【深入学习】代码中<ul>标签内的部分，即第 7～15 行是一个简单的列表。其中罗列了 4 个条目，分别是姓名、年龄、性别、爱好，每个条目之间用<p>标签空一行，使每一条目之间空开一行，不至于显得太拥挤。在默认情况下，无序列表的条目符号是实心的黑色小圆圈，如图 3.13 所示是这个登录信息的页面在浏览器中显示的结果。

3.4.2　有序列表

有序列表中的条目依次按照顺序排列。有序列表和无序列表之间的唯一区别体现在代码上，即有序列表使用
标签，以
开始，到
结束。有序列表中同样使用<li>标签来定义列表的每一行，具体的写法如下：

```
<ol>
  <li>……</li>
  <li>……</li>
  <li>……</li>
</ol>
```

只要在程序 3.12 中做一些小小的改动，将代码第 7 行和第 15 行的<ul>标签替换成<ol>标签。那么，原先的无序列表就变成有序列表的样式了，如图 3.14 所示就是有序列表的样式，每个条目依次排列数字。



图 3.14　有序列表

3.4.3　定义列表

定义列表是一种缩进样式的列表，设计的本意是要用于定义术语。代码中使用<dl>来创建定义列表。在列表中使用<dt>来定义页面中的每一行。与有序列表和无序列表不同的是，在定义列表中，列表中会添加缩进行来展示这个列表的条目，使用<dd>标签来定义缩进行。它的代码写法如下：

```
<dl>
  <dt>…</dt>
  <dd>…</dd>
  <dt>…</dt>
  <dd>…</dd>
</dl>
```

现在使用定义列表举一个简单的例子，如程序 3.13 所示为使用定义列表来表现一段对于名词解释的页面，在图 3.15 中，可以看到这个定义列表在浏览器中的效果。

【本节示例参考：资料光盘\第 3 章\3-13　制作定义列表.html】。

【实例 3-13】制作定义列表的方法，其源码展示如下：

程序 3.13　制作定义列表.html

```
01  <html>
02    <head>
03      <title>制作定义列表</title>
04    </head>
05    <body >
06      <h3>镜头画面的剪辑</h3>
07         <dl>
08            <dt>分剪</dt>
09            <dd>一个镜头分成两个镜头或者两个以上的镜头使用。</dd>
10            <dt>挖剪</dt>
11            <dd>将一个完整镜头中的动作、人和物运动镜头在运动中的某一部位上的多
12  余的部分挖剪去。</dd>
13            <dt>拼剪</dt>
14            <dd>将一个镜头重复拼接。</dd>
15          </dl>
16    </body>
17  </html>
```

【运行程序】浏览该页面，结果如图 3.15 所示。



图 3.15　使用定义列表

【深入学习】在定义列表中，放在<dd>标签中的代码会作为缩进行显示在浏览器中。此外，许多页面设计者使用<dl>和<dd>标签用来替代<blockquote>标签的功能，以此来缩进文本行。即在需要缩进的文本前面加<dl><dd>，在文本结束的地方加上</dl></dd>，其作用和<blockquote>标签是一样的，该标签使内容缩进而不将其视为定义。

3.4.4　列表嵌套

通常，在设计页面使用列表时经常会遇到将一个列表放入另一个列表中的情况，以一个列表看作一个新列表中的一行条目，这种情况称之为列表嵌套。列表嵌套就好像在一个大盒子中放入一个较小一些的盒子，在较小一些的盒子中再放入一个小盒子，总之就是盒子装着盒子，列表嵌套就是列表里面还有列表。

无论是无序列表嵌套，还是有序列表嵌套，或者是无序列表和有序列表的混合列表，它们的代码写法都是一个原则，就是遵从 HTML 代码的使用规则，将一个列表的标签完全放在另一个标签内，这

是一种父子级的关系。如程序 3.14 中使用了列表嵌套来表现书籍的目录，这种方法常用来表示复杂的导航，应用广泛。

【本节示例参考：资料光盘\第 3 章\3-14　列表的嵌套.html】

【实例 3-14】 列表的嵌套，其源码展示如下：

程序 3.14　列表的嵌套.html

```
01   <html>
02     <head>
03       <title>列表嵌套</title>
04     </head>
05     <body>
06       <h3>四大名著</h3>
07     <ul style="list-style-type:disc">                    //定义列表的项目符号
08       <li>列表一
09         <ul style="list-style-type:circle">
10           <li>《三国演义》
11               <ul style="list-style-type:square">
12                   <li>第 一 回   宴桃园豪杰三结义 斩黄巾英雄首立功</li>
13                   <li>第 二 回   张翼德怒鞭督邮 何国舅谋诛宦竖</li>
14                   <li>第 三 回   议温明董卓叱丁原 馈金珠李肃说吕布</li>
15               </ul>
16           </li>
17           <li>《水浒传》
18               <ol>
19                   <li>第 一 回   张天师祈禳瘟疫 洪太尉误走妖魔</li>
20                   <li>第 二 回   王教头私走延安府 九纹龙大闹史家村</li>
21                   <li>第 三 回   史大郎夜走华阴县 鲁提辖拳打镇关西</li>
22               </ol>
23           </li>
24       </ul>
25       <ol style="list-style-type:upper-roman">
26           <li>《西游记》
27               <ul>
28                   <li>第 一 回   灵根育孕源流出   心性修持大道生</li>
29                   <li>第 二 回   悟彻菩提真妙理   断魔归本合元神</li>
30                   <li>第 三 回   四海千山皆拱伏   九幽十类尽除名</li>
31               </ul>
32           </li>
33           <li>《红楼梦》
34               <ol style="list-style-type:upper-alpha">
35                   <li>第 一 回   甄士隐梦幻识通灵 贾雨村风尘怀闺秀</li>
36                   <li>第 二 回   贾夫人仙逝扬州城 冷子兴演说荣国府</li>
37                   <li>第 三 回   托内兄如海荐西宾 接外孙贾母惜孤女</li>
38               </ol>
```

```
39            </li>
40          </ol>
41        </ul>
42      </body>
43    </html>
```

【运行程序】这个书籍目录的页面在浏览器中的显示结果如图 3.16 所示。

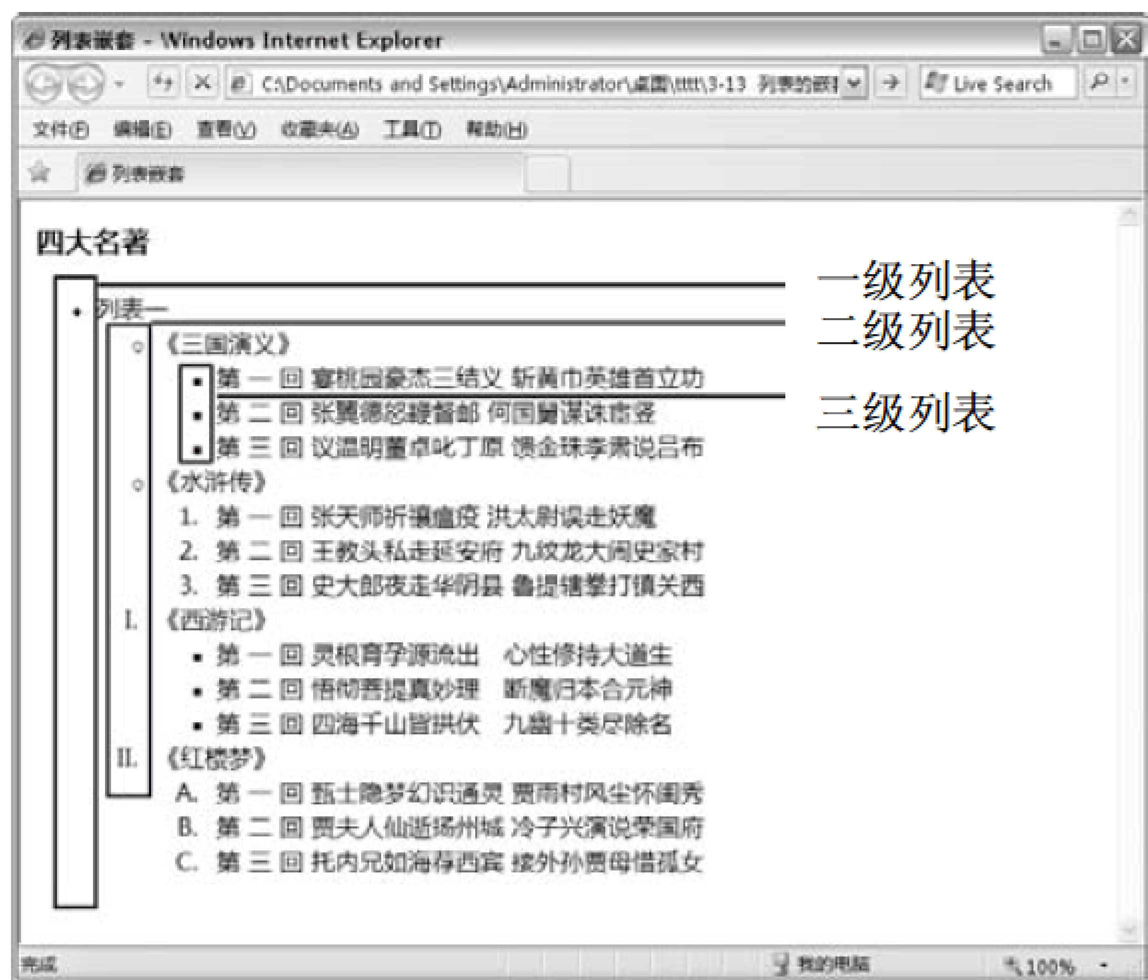


图 3.16　列表的嵌套

【深入学习】该页面中包含了多级列表的嵌套，共有 3 层关系，最外层的是一级列表，虽然它只有一个列表行，即条目“列表一”，体现在程序 3.14 中第 7～41 行。而一级列表的条目下是一个新的无序列表和一个新的有序列表，如代码中第 9～24 行是无序列表部分，第 25～40 行是有序列表部分，它们是二级列表。

在无序列表和有序列表下，分别有两个列表条目，无序列表下有条目《三国演义》和《水浒传》，有序列表下有条目《西游记》和《红楼梦》。在每一个书名目录下，在示例中用列表列举了章回目录，这些列表则是全局的三级列表。代码中第 11～23 行是无序列表下嵌套的列表部分。从第 27～39 行是有序列表下嵌套的列表部分。

在列表嵌套时，不同级的列表下的条目符号都是不同的。默认情况下，无序列表中一级列表是黑色实心小圆，二级列表是空心小圆，三级列表是黑色实心小方块。有序列表无论是哪一级的列表，一般都是以阿拉伯数字为条目符号。当然，如果设计者改变条目符号的样式，可以通过代码定义不同的样式，条目符号可以通过添加“style="list-style-type:….."”属性来改变。

如代码第 7 行、9 行、11 行、25 行、34 行中，disc 是实心黑色的小圆圈，circle 是空心的圆圈，square 是实心黑色的小方块，它们属于无序列表。upper-roman 是大写的罗马数字，upper-alpha 是英文大写字母，此外还有属性 lower-alpha 是小写字母，lower-roman 是小写的罗马数字。本例正是通过这样的方式来改变列表条目符号的样式。

3.5　制作一则通知

本节结合前面所学的知识，通过制作一则会议通知来整体理解本章的知识点。首先，制定好页面的内容，一则会议通知需要包括通知的标题、通知的内容，还有最后的署名。

既然分为 3 个部分去完成，就可以一次针对每一块需求来编写页面代码，最后再将它们整合在一起编辑格式。这个页面的 HTML 文档如程序 3.15 所示。

【本节示例参考：资料光盘\第 3 章\3-15　一则会议通知.html】。

【实例 3-15】制作一则会议通知，其源码展示如下：

程序 3.15　一则会议通知.html

```
01  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03  <html xmlns="http://www.w3.org/1999/xhtml">
04   <head>
05     <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
06     <title>制作会议通知</title>
07   </head>
08  <!—以下是页面的主体部分 -->
09   <body>
10     <h2 align="center">关于________会议的通知   </h2>
11       <p>各职能处室：</p>
12       定于×月×日召开××××会。现将有关事宜通知如下：
13       <br><pre>
14     <ul><p><li>   会议议题：________________________
15           <p><li>   参加人员：________________________
16              <br>            ________________________
17              <br>            ________________________
18           <p><li>   会议时间：从___到___结束
19           <p><li>   会议地点：________________________
20           <p><li>   具体事项：
21              <ol><li>   __________________
22                  <li>   __________________
23                  <li>   __________________
24              </ol></pre>
25         </ul>
26     <p align="right">______________公司
27     <p align="right">         年     月     日
28   </body>
29  </html>
```



注意：因为这个代码中使用了<pre>标签来控制排版格式，所以在编写代码时，要特别注意格式的工整。

【运行程序】这段代码最终在浏览器中的显示结果如图 3.17 所示。

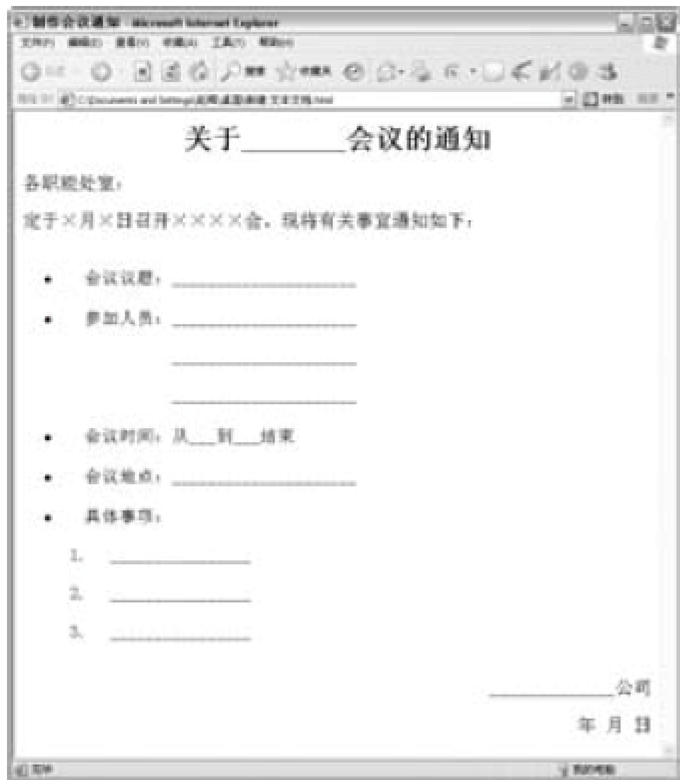


图 3.17　制作一则通知

【深入学习】代码第 10 行定义了通知的标题，并且使其居中显示。代码的第 11～25 行是通知内容的正文部分，设计者使用了一个无序列表来罗列通知的注意事项。从第 14～25 行是这个无序列表，其中在描述“具体事项”时，设计者又在这个无序列表中嵌套了一个有序列表，用来罗列“具体事项”的条目。第 21～24 行代码是一个有序列表，最后第 26 行和第 27 行代码是通知署名的部分，并使之右对齐显示。

如果将这个页面上传到互联网上，便可以提供给需要者下载使用，或者是作为表格样板直接用来打印，这样就避免了每次都要从“我的电脑”中搜寻同样版式的表格。

3.6　小　　结

本章主要学习了文本的编排，通过本章的学习，读者可以练习设计纯文本的页面。其中主要的知识点有：

- ❑ 使用<p>或者
进行文本的换行。
- ❑ 使用标签改变页面字体的大小。
- ❑ 使用对齐属性改变页面的排版。
- ❑ 一些重要的特殊标签，如空格符号、<pre>和<acronym>标签等。
- ❑ 在页面中使用列表，以及如何运用列表的嵌套。
- ❑ 最后，通过制作一则通知展示了本章所学知识的魅力。

当然，一个页面中仅有文本是不够的，在后面的章节中，将学习如何编辑页面中的图像，使设计出来的页面更加丰富多彩。

第 4 章　将图像放入页面中

一个页面中，除去文本部分，最常见的文件就是图像了。现实世界中，人们所指的图像最常见的莫过于相片或画，专业照片是记录在胶片上的。画当然是描绘在画布或画纸上的，而通常所说的数码相机记录下来的数字照片，是存放在存储卡上的。存放下来的这些图像可以通过不同的浏览器被阅览，而这种图像才是我们所要关注的图像。在网页中，一张图像可以大到覆盖整个页面的背景，也可以小到微不足道的一个 logo，它们在不经意间充斥着整张页面。如图 4.1 所示是 YAHOO 门户网站的一张主页。



图 4.1　充斥着图像的页面

YAHOO 主页中，图像所占的面积甚至和文本所占的面积相当。这样，才会让页面显得丰富、直观、不呆板、吸引浏览者的眼球。本章的知识点如下。

- 计算机图像的概念。
- 如何用计算机术语表达图像。
- 学习如何排版图像。
- 制作自己的图像。
- 使用技巧来美化图像。
- 设置页面背景。

4.1　图像的基础知识

页面中放入图像不难，使用 HTML 标签很容易就可以做的，困难的是明白放入什么格式的图像，

如何去修改图像，如何在网页中去应用图像，这些知识基于需要对图像知识有个清晰的认识。了解这些，能使一个设计师在制作页面时，更加游刃有余、得心应手。

4.1.1 最常用的图像——位图

位图又称为光栅图，是由许多像素组成的图，像素是很小的颜色块，如马赛克一样，试想一下用小瓷砖拼成的卫生间地板，位图就是这个样子。如图 4.2 所示是一张被放大的图像，当图像中模型的头部放大到 700 倍时，图像呈现出锯齿一样的边缘。当图像放大到 1400 倍时，看到的图像只有一个一个的小方格，这每一个小方格，就是像素。像素表示为一个正方形的颜色块。

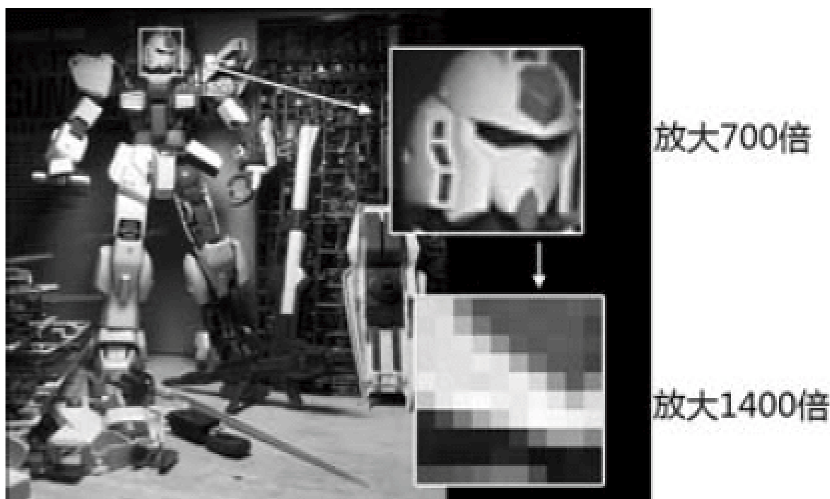


图 4.2 观察图像的像素



注意：放大原始位图，能使图像效果失真，如果缩小原始位图，同样会使图像效果失真，这是因为图像的缩小，减小的是图像中像素的数量。

位图通常又被分为 8、16、24、32 位图。这并不是说只有 8 种颜色，而是有 2 的 8 次方，即 256 种颜色。从人眼的感觉来说，16 位色基本能满足需要了。而 24 位色又被称之为“真彩色”，2 的 24 次方大概是 1677 万之多，这个数字差不多是人眼可以分辨颜色数的极限。而 32 位色是个例外，并不是说有 2 的 32 次方之多的发光数，它是在 24 位色的基础上加入了 256 阶颜色的灰度。

在制作页面时，设计者一般选择 24 位图像，32 位图像虽然质量更好，但同时也带来更大的图像容量。如果一个页面使用体积过大的画，会导致浏览者浏览页面的拖慢，而事实上，一般肉眼也很难分辨 24 位图和 32 位图的区别。

4.1.2 在页面中常用的位图格式

生活中，画的种类可以分很多，素描、油画、水墨画等。在计算机的世界里，图像也可分为很多种格式，但问题是，人们凭借肉眼就可以在一堆画中分辨出素描和油画，却不可能凭借肉眼直观地分辨计算机世界的图像，它们看上去都是一样的。虽然人眼做不到，而我们的帮手计算机则可以。计算机可以把不同的图像定义为不同的格式，不同格式的图像有它自己的属性和特点，而人们只要了解了不同格式图像的属性和特点，就能掌握运用图像的门道。在页面中，常用的 3 种位图图像格式分别是 JPEG 图像、PNG 图像和 GIF 图像。

1. JPEG 格式的图像

JPEG 文件是最常见的一种图像格式之一，后缀名是.jpg，几乎所有的软件或平台都可以打开它，

JPEG 文件是压缩过的一种图像。压缩的图像可以保持为 8 位、24 位、32 位深度的图像，压缩比率可以高达 100∶1。JPEG 可以很好地处理大面积色调的图像，如摄影作品、相片。但它不适用于色彩对比强烈、有清晰边界、简单构图的图像，如 logo、banner。

2．PNG 格式的图像

PNG 文件的后缀名是.png。这是一种能够存储 32 位信息的位图图像，采用的是一种无损压缩的方式。目前，已经在 Web 上广泛流行。此外，它支持透明信息。所谓透明，即是说图像可以浮现在其他页面文件或页面图像之上。可以说，PNG 文件是专门为 Web 创造的图像，通常，大部分页面设计者在页面中加入 Logo 或一些点缀的小图像，都会选择使用 PNG 文件。

3．GIF 格式的图像

GIF 是一种图像交互格式，后缀是.gif，它只支持 256 色以内的图像。所以，GIF 文件的图像效果是很差的。但是，GIF 文件有一个最大的特点，就是可以制作动画，图像制作者利用图像处理软件，将静态的 GIF 图像设置为单帧画面，然后把这些单帧画面连在一起，设置好上一幅画面到下一幅画面的间隔时间，最后保存为 GIF 格式就可以了。可以说，这就是简单的逐帧动画。目前这种格式的动画在互联网上广为流行。

总的来说，这 3 种图像在使用上各有千秋，JPEG 可以压缩图像的容量，PNG 的质量较好，GIF 可以做动画，所以，当处理色调复杂、绚丽的图像时，如照片、图画等，适合使用 JPEG；而处理一些 logo、banner、简单线条构图时，适合使用 PNG；而 GIF 通常只是用来表达动画效果。

当然，位图图像的格式还有很多，如 BMP、TGA、TIFF、PSD 等，这些在页面中并不常用。这些格式的图片容量都较大，通常，图像设计者为了保证图像的质量，使用这些格式的图片进行设计修改，最后成品仍需要转换为网页中常用的图像格式。



注意：对于图像使用，并没有特别严格的要求，设计者可以依照自己的习惯选择使用不同格式的图像。

4.1.3　奇妙的矢量图

计算机中显示的图形一般分为两大类，前面章节中介绍的是位图，这是其中一种，而另一种就是矢量图。矢量图和位图最大的区别在于，前者无论图像大小的缩放，都不会影响其效果，而后者会有损图像质量。如图 4.3 所示，把“小乌龟”眼部放大之后可以看到，“眼睛”部位的边缘并没有失真。

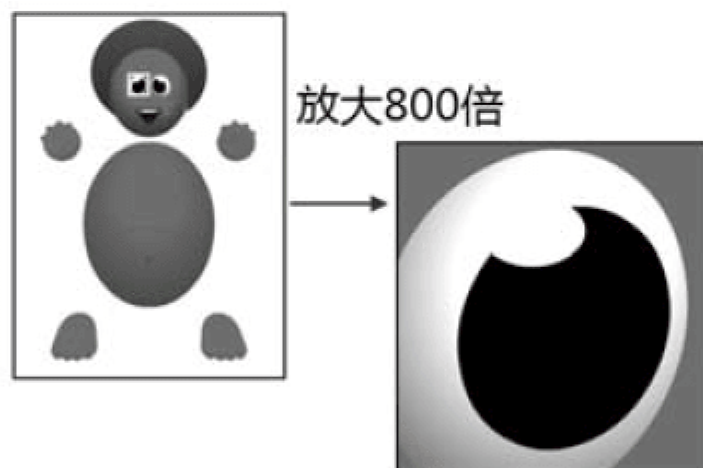


图 4.3　放大矢量图

矢量图是以一种数学描述的方式来记录图像内容。例如，一个方程 y=kx，当这个小方程体现在坐标系上时，设置不同的参数可以绘制一条任意角度的直线，这就是矢量图的构图原理。矢量图一般以线条、曲线和色块为主，不易制作色调丰富、效果绚丽的图像，其文件所占的容量也比较小。

矢量图由于其可以随意放大缩小的特点，通常被看作是一种图像模板放在图库中，方便设计者备用保存，在需要时拿出来使用。遗憾的是，矢量图不易制作色调复杂图像的因素，这一局限迫使设计者一般只愿意将 logo、UI 图标、标识符号等简单图像存为矢量图。

矢量图的后缀一般有.ai、.cdf、.fh、.swf 等。.ai 后缀的文件是一种静帧的矢量文件格式，.cdf 后缀的文件多以工程图为常见。而.swf 格式文件其实指的就是 Flash，Flash 也是页面中最常见的一种动画，将在后面的章节中介绍。

4.1.4　图像的分辨率

分辨率的单位是 dpi（display pixels/inch），即每英寸显示的线数。通常人们所指的分辨率有两种，分为屏幕分辨率和图片分辨率，屏幕分辨率即指计算机显示器默认的分辨率。

一般来说，目前大部分显示器的分辨率是 1024×768。即是说，如果设计者制作的页面范围超过了显示器默认的分辨率，那么即使是在浏览器全屏的情况下，页面也不能完全在浏览器中展示出来，所以这种情况下，浏览器中就出现了滚动滑条。

图像的分辨率是用于量度位图图像内数据量多少的一个参数。分辨率越高的图像，包含的数据越多，图像的容量也大，需要消耗更多的计算机资源，需要更大的存储空间。所以，对于页面开发者，选择适当的图片才是最好的选择。



注意：并不是说图像分辨率越高的图像就一定越清晰，这其中还要看图像本身制作的水准。

那么，像素和分辨率又有什么关系呢？分辨率指的是每英寸的像素值，通过像素和分辨率的换算可以测算长度。举例来说，一幅 400×300 分辨率的图像，在一个屏幕分辨率为 300dpi 的浏览器中，最终图像的长和宽是通过分辨率和像素的换算得出的。

1 英寸=2.54 厘米。

400/300×2.54=3.39 厘米。

300/300×2.54=2.54 厘米。

那么最终显示的是一个长约 3.39 厘米、宽 2.54 厘米的图片，所以，当遇到不同屏幕分辨率的显示器时，换算成厘米的数值也是不同的。

4.1.5　认识一些网页中常用的 banner 尺寸

在网页中，经常涉及使用 banner，banner 即是指网幅广告、旗帜广告、横幅广告等。一个优秀的页面，不仅是放入的 banner 要美观精致，同时更要符合页面的风格，banner 有一定的格式标准可供参考，但不是说一定要完全严格遵循。几种国际尺寸的 banner 如下。

- ❑ 468×60：全尺寸 banner，应用最为广泛的广告条尺寸，用于页眉或页脚。
- ❑ 392×72：全尺寸带导航条 banner，主要用于有较多图片展示的广告条，用于页眉或页脚。

- 234×60：半尺寸 banner，这种规格适用于框架或左右形式主页的广告链接。
- 125×125：方形按钮，这种规格适于表现照片效果的图像广告。
- 120×90：按钮类型，主要应用于产品演示或大型 logo。
- 120×60：按钮类型，这种广告规格主要用于做 logo 使用。
- 88×31：小按钮，主要用于网页链接，或网站小型 logo。
- 120×240：垂直 banner。



说明：读者可以依照这些参考数据，裁定自己放入页面中的 banner 大小。

4.2　用图像来编织美丽的页面

图像是增加网页可视效果的最有力武器之一，在语言文字无法描述的地方，一些只可意会无法言传的表达中，也许设计者使用一张图片就可以完美传达设计者的信息。本节将介绍如何在页面中放入图像、编辑图像。

4.2.1　理解图像路径

在页面中放入图像，实际上是使用了服务器中的一张图片。设计者把所有的图片放在一台服务器的某个文件夹中，然后通过 HTML 语言告诉浏览器这些图片放在哪里，即图片的路径，这就好比是图片住在计算机中的门牌号码。当用户浏览页面时，浏览器会解析这张图片，通过获知的“门牌号码”找到这张图片。在页面文档中，使用<img/>标签将图像放入页面中，使用的格式如下：

```
<img src=… alt=… />
```

img 表示 image，src 表示 source，即图片的源。标签中，src 属性用来指定图像的位置，alt 属性指定关于图像的描述性文本。如果浏览者不能看到图像，将看到 alt 属性注释的文本。如程序 4.1 在页面中的这幅图像。

【本节示例参考：资料光盘\第 4 章\4-1　在页面中添加图像.html】

【实例 4-1】在页面中添加图像的方法，其源码展示如下：

程序 4.1　在页面中添加图像.html

```
01        <html>
02          <head>
03             <title>在页面中添加图像</title>
04          </head>
05          <body style="text-align:center">                //令文本居中显示
06        <h3>向左走 向右走
07             </h3>
08             <p><img src="图片/向左向右.jpg" alt="向左走向右走"/>
```

```
09              </body>
10          </html>
```



注意：<img/>标签中的斜杠如果省略也可以，但是 XHTML 标准要求任何空标签，即没有结束标签的标签，都要在结尾包含一个斜杠。

【运行程序】最终在页面中的显示结果如图 4.4 所示。



图 4.4　在页面中添加图像

【深入学习】如代码中第 8 行，src 属性下指向“图片”文件夹中的“向左向右”这张 JPEG 图片。alt 属性下是对这张图片的注释，如果浏览者未能看到图片，将能看到设计者对图片的注释。

4.2.2　像编辑文本对齐一样在页面中对齐图片

如果在一个页面中放入图片，可以像编辑文本一样令图片左对齐、右对齐或居中对齐，只要在<img/>标签中加入 align 属性即可，如程序 4.2 的使用案例。

【本节示例参考：资料光盘\第 4 章\4-2　在页面中对齐图片.html】

【实例 4-2】在页面中对齐图片的方法，其源码展示如下：

程序 4.2　在页面中对齐图片.html

```
01   <html>
02          <head>
03                 <title>在页面中对齐图片</title>
04          </head>
05          <body>
06                 <h4>"The Shot"</h4>                              //这里是文本的标题
07                  <br>
08                 1989 年季后赛首轮第五场，Michael 在最后 3 秒接球、运球、起跳…
09                     <p align="center"><img src="图片/表情 2.jpg" />   //使图像居中对齐
10               </p>
11                     <p ><img src="图片/表情 1.jpg" align="left" />      //使图像居左对齐
12                           <img src="图片/表情 3.jpg" align="right"/>     //使图像居右对齐
13            </body>
14     </html>
```

【运行程序】浏览该页面，结果如图 4.5 所示。



图 4.5　页面中图片的对齐属性使用

【深入学习】如本例中第 11、12 行，图像的对齐和文本的对齐在使用代码上略有不同，当对图像命令左对齐和右对齐时，align 属性可以放在<img/>标签内，也可以放在<p>标签或其他标签内，而居中对齐的命令放在<img/>标签内不起作用，图片依然是左对齐。正如图 4.5 在浏览器中的显示效果那样。

同时，第 11、12 行中的属性命令实际上还针对和图像并列的文本内容，可以理解为，使图像放在文本的左边，或者是使图像放在文本的右边。所以，从这个角度就可以理解，为什么属性中没有居中对齐的使用命令。HTML 语言活用的方法有很多，这些经验需要慢慢积累。



注意：目前，在设计页面版块时，通常在放置图片前会先设计好表格、框架或使用层，这些在后面的章节中会继续介绍，而这种放置图片的方法是学习 HTML 语言的基础，并不实用。

4.2.3　图像与文本的对齐方式

在编辑图像时，图像不同于文本，图像都是一个个分开的整体。而编辑图像时，如果设计者想在图片的旁边放入文本内容，就需要考虑如何处理文本和图像对齐方式。在 HTML 文档中分为如下 4 种。

- 使图像的顶部和同一行的文本对齐，使用代码如下：

```
<img style="vertical-align:text-top"/>
```

- 使图像的中部和同一行的文本对齐，使用代码如下：

```
<img style="vertical-align:middle"/>
```

- 使图像的底部和同一行的文本对齐，使用代码如下：

```
<img style="vertical-align:text-bottom"/>
```

- 使图像的底部和文本的基线对齐，使用代码如下：

```
<img style="vertical-alignbaseline"/>
```

程序 4.3 中，分别使用了 text-top、middle 和 baseline 这 3 种对齐样式。

【本节示例参考：资料光盘\第 4 章\4-3　图像与文本的对齐方式.html】

【实例 4-3】图像与文本的对齐方式，其源码展示如下：

程序 4.3　图像与文本的对齐方式.html

```
01        <html>
02          <head>
03             <title>图像与文本的对齐</title>
04          </head>
05          <body>
06            <h4>图像与文本的对齐方式</h4>
07             <p><img src="图片/图鉴 1.jpg"                          亡灵掌握着许多威力
08        十足的魔法，包括操纵死去的战士作战的恐怖魔法。
09               <!—文本和图像顶部对齐  -->
10   <p><img src="图片/图鉴 2.jpg" style="vertical-align:middle"/>兽人类似人族，但拥有巨大的
11   力量和颇高的生命值。
12               <!—文本和图像中间对齐  -->
13   <p><img src="图片/图鉴 3.jpg"   style="vertical-align:baseline"/>巨魔猎手用嗅觉灵敏能力
14        追踪宿敌，它们更倾向于在黑暗中猎杀对手。
15               <!—文本和图像底部对齐  -->
16          </body>
17        </html>
```

不同样式的对齐方式

【运行程序】浏览该页面，结果如图 4.6 所示。

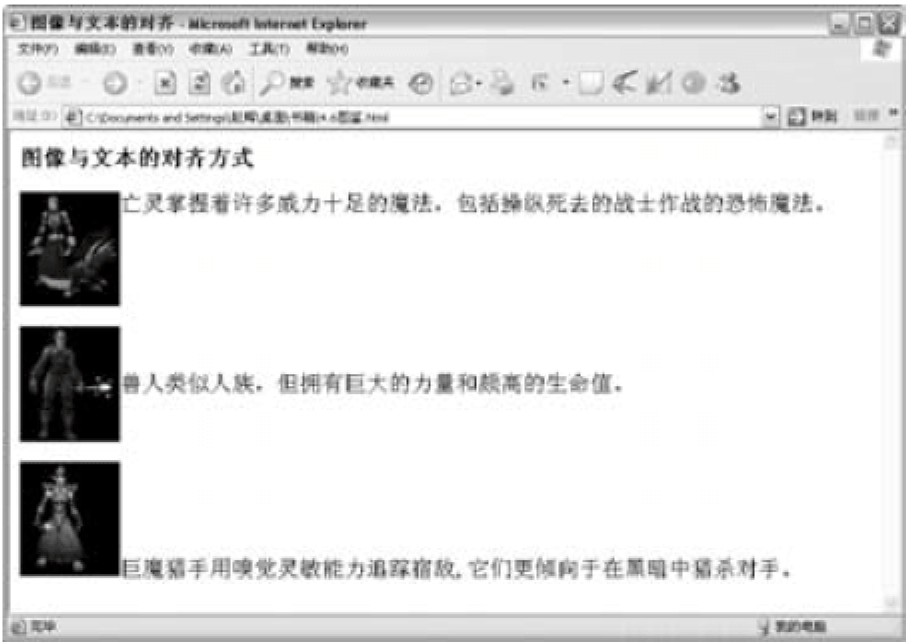


图 4.6　图像与文本的对齐方式

> **说明：baseline 属性的效果和 text-bottom 属性几乎相同，后者文本不会超出图片的下边线，可参考图中 text-top 属性的样式，文本最上线没有超过图像上线。**

4.2.4　控制图像与文本的距离

编辑页面时，除了可以控制图像与文本的编排方式外，甚至可以进一步调整图像和文本的距离。

当设置好图片和文本的距离后，可利用 hspace 和 vspace 属性来分别控制图像四周与其他内容间隔的水平方向的宽度，或者是垂直方向的高度，这种效果可以令页面展示出更多不一样的特色。具体案例的使用如程序 4.4 所示。

【本节示例参考：资料光盘\第 4 章\4-4　控制图像与文本的距离.html】

【实例 4-4】控制图像与文本的距离的方法，其源码展示如下：

程序 4.4　控制图像与文本的距离.html

```
01        <html>
02          <head>
03            <title>图像与文本的距离控制</title>
04          </head>
05          <body>
06             <h4>图像与文本的距离控制</h4>
07             <img src="图片/黑白照片.jpg" hspace=30 >这段文字距离左边图像的距离是 30px。
08           <p><img src="图片/黑白照片.jpg" vspace=30 ><br>这段文字距离上边图像的距离是
09   30px。
10          </body>
11   </html>
```

【运行程序】从图 4.7 中可以看出这一效果。



图 4.7　控制图像与文本的距离



注意：左右两边的距离都是 30px，而下一幅图中，上下两边的距离都是 30px。

【深入学习】在设置距离数值时，使用的标准单位是像素，英文表示缩写即 px。hspace=30 是指控制图像左右两边与页面其他内容间隔 30px 的距离。同样，vspace=30 是指图像上下两边与页面其他内容的间隔距离是 30px。

4.3　让图像变得更美观

如果使用 Google 或 Baidu，设计者可以找到任何自己需要的图像，可是这些图像又不可能完全符合设计者的想法，每个设计者总要面对改变图像的大小、截选图片的部分区域等问题，在本节中，将结合 Windows 自带的画图工具，介绍一些基本的修改图像的方法。

4.3.1　使用画图工具修改图像

在 HTML 文档中有一种可以编辑图像的大小的方式，在<img/>标签中添加“宽”和“高”的属性，如程序 4.4 中第 7 行如果改成：

```
<img src="图片/黑白照片.jpg" width="300" height="100">
```

显示的结果如图 4.8 所示。原始图像将整体被压缩变形。虽然使用代码改变了图像的大小，得到了需要规则的宽和高。但是，这种方法也使图像遭到严重的变形，如图 4.8 中最上面的图。所以很少有人愿意在页面文档中直接编辑，通过外部软件来修改图像比较方便，这样的软件很多，最常见的如 Photoshop、Firework 等。这里对于基础的修改图像，只需要用 Windows 自带的画图软件就可以做到了。如图 4.9 所示为电影《Love Actually》的一张海报。





原始图

图 4.8　失真对比



图 4.9　Love Actually

很显然，这么一幅大海报是不适合使用的，现在希望将海报中间的“love actually”取出来作为页面的某个图标。那么首先单击桌面上的“开始”按钮，在弹出的菜单中选择“附件”→“画图”命令，打开“画图”工具。在其中打开这张图，如图 4.10 所示。



图 4.10　如何截取图片

使用左边工具栏第一排的右边工具，即截选图片工具，选出需要的部分，如图中右边放大的样式。这部分图中，如果不想保留图中的“are in”字样，可选择工具栏第二排左边的“橡皮擦”工具，可以擦去“are in”字样，然后重新选取需要的部分图片并右击，在弹出的快捷菜单中选择“剪切”命令。

之后，在顶栏的“文件”菜单下选择“新建”命令，创建新的图像。在新的图像中，保持左侧工具栏在“截选图片”工具选择下，在弹出的快捷菜单中选择“粘贴”命令，最后通过拖拽图像的边缘修改图像的大小。完成以后，在“文件”菜单下选择“保存”命令或“另存为”命令即可，如图 4.11 所示。



图 4.11　修改完成的图像

画图工具的功能不多，只能做一些最简单、最基本的修改图片的功能，市面上修改图像的软件数不胜数，最赋盛名的可能就是 Photoshop 了，本书会在动感页面篇中讲到 Photoshop 的使用。



说明：Windows 自带的画图工具功能有限，如果希望制作更多效果的图像，Photoshop 是一个很好的选择。

4.3.2　不一样的改变——给图像添加边框

编辑图像时，有一种使用频度很高的修饰图片的方式。给图像添加边框，虽然这是对图片小小的修饰，但带来的效果是相当突出的。在<img/>标签中添加 border 属性，它的写法如下：

```
<img src="..." border=   >
```

其中，在 border 属性下输入像素值，指边框的宽度。在程序 4.5 中，用这个方法给先前的截图添加了边框。

【本节示例参考：资料光盘\第 4 章\4-5　给图像添加边框.html】

【实例 4-5】给图像添加边框的方法，其源码展示如下：

程序 4.5　给图像添加边框.html

```
01          <html>
02            <head>
03              <title> 给图像添加边框</title>
```

```
04              </head>
05              <body>
06                <h4>添加图像边框</h4>
07                <p align=center>
08                <img src="图片/la 图标.bmp" border=8 >           //给图像添加边框
09                 </p>
10              </body>
11            </html>
```

【运行程序】在页面中显示的效果如图 4.12 所示。



图 4.12　添加图像边框

【深入学习】这是前面小节中，从电影海报中截取出来的图像。这幅图像添加了边框之后，立刻在页面中显得更突出，更容易抓住人的眼球，使图像也有了一种个性化的标示。如果读者掌握了 CSS 的规则，甚至可以改变图像边框的颜色，修改图像的边角。而这些，本书将在后面的篇章中具体介绍。



注意：这里修饰的做法并没有将结构和表现分离。当了解了 CSS 样式表之后，可以使用一种更好的方法。

4.3.3　独树一帜的水平线

在设计页面时，经常需要在网页中插入一条水平线来隔开文本，或者是为了起到美化页面的作用，水平线是设计页面中的一个特殊的小部分，使用页面标签可以实现这个功能，代码的写法如下：

```
<hr align="…" width="…" size="…">
```

<hr>标签即是放入水平线的意思。在编辑水平线时，可以使用 align 属性编辑其对齐模式。width 属性和 size 属性下填入具体的数字，单位是像素。width 属性即表示水平线的长度，而 size 属性用来表示水平线的宽度。



说明：在本章 4.5 节中的案例将使用到水平线。

4.4　改变页面的背景

放入背景图片的方式和插入图片的方式是不同的，添加图可以放入在页面中任意位置，但不能在插图上编辑页面内容，而背景图像就是整个页面的最底层，原先是一片空白的页面改成了一张图片而已，设计者可以在背景图像上放入任何页面内容，使用的页面代码如下：

```
<body background=
 background-repeat: …">
```

在页面主体开始的<body>标签中使用样式 background-image 属性，即背景图像属性，用来指定背景的图片，使用时在 url 后面放入指定的图片地址。background-repeat 属性用来定义背景图像在水平方向或垂直方向上重复使用，以免因为图像太小而不能铺满整个页面，水平方向重复即添加 repeat-x，垂直方向重复即添加 repeat-y。这种方法好比在地板上铺瓷砖，每一幅背景图看作是一块瓷砖，而需要重复使用很多这样的瓷砖来铺满地板。程序 4.6 中演示了这一方法的使用。

【本节示例参考：资料光盘\第 4 章\4-6　设置页面背景图像.html】

【实例 4-6】设置页面背景图像的方法，其源码展示如下：

程序 4.6　设置页面背景图像.html

```
01          <html>
02            <head>
03              <title> 设置页面背景图像</title>
04            </head>
05          [                                        ]  ◄── 背景图像的路径
06                              background-repeat:no-repeat">
07                   <br><h3><p style="color:#ff0000">添加页面背景
08            </body>
09          </html>
```

【运行程序】在浏览器中显示的效果如图 4.13 所示。



图 4.13　添加页面背景

【深入学习】代码第 5 行引用了一张命名为“页面背景”的图片，注意这一行最后的分号“;”，包括 HTML 中所有的标点符号，一定要在英文输入法下输入。



说明：代码第 7 行中，<p>标签中的 style 属性中“color:#ff0000”修改文本的颜色，这种用法会在后面的章节中介绍。

4.5　案例：把宠物的照片放到网页上去

本章涉及了很多编辑图像的技法，本节通过运用这些技法将这些技能运用到实际操作中去。很多人现在有自己的宠物，有些人会把自己宠物的照片晒到互联网上，给自己的宠物找玩伴，藉此来认识同样喜爱宠物的朋友。在这个例子中，将学习如何把自己宠物的照片晒到互联网上。

首先，需要对这个宠物页面给出一个设计方案，明确基本的构思。如在这个例子中，需要 3 个设计要点。

- ❑ 一个页面的 banner。
- ❑ 宠物的图像要放在页面的右侧，应该给图像加上边框。
- ❑ 定义页面的文本内容，放在页面的左侧。

当设计者规划好这 3 部分之后，就逐一放到代码文档中去，如程序 4.7 所示。

【本节示例参考：资料光盘\第 4 章\4-7　我的宠物页面.html】

【实例 4-7】制作“我的宠物”页面，其源码展示如下：

程序 4.7　我的宠物页面.html

```
01          <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03          <html xmlns="http://www.w3.org/1999/xhtml">
04            <head>
05               <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
06               <title>我的宠物</title>
07            </head>
08
09            <body>
10             <p><img src="图片/我的宠物 banner.gif" width="1017" height="24" /></p>
11        <!—这里放入 banner，长度是 1017px 高度是 24px -->
12             <p>
13             <p>
14             <p><h2>Jenny</h2>
15             <p><img src="图片/我的宠物.gif" width="192" height="262" border="2" align="right"
16    /></p>                                           //放入图像
17             <p>年龄：24<br/>
```

```
18                  最喜爱的食物：小虾、肉<br/>
19                  特点：很安静</p>
20              <p><hr align="left" width="400" size="3" >          //放入水平线
21                  <p>  更新宠物日记</p>
22                  <hr align="left" width="400" size="3" >          //放入水平线
23                  <p>2008 年 9 月 1 号</p>
24                  <p>  外面的天气还是很热，所以 Jenny 一直都懒懒的趴在那里，好久也…
25              </body>
26          </html>
```

【运行程序】这个案例最后在浏览器中的显示效果如图 4.14 所示。



图 4.14　我的宠物

【深入学习】页面主体中，即<body>标签内，其中第 10 行代码表示放入页面的 banner。第 15 行表示放入页面中宠物的照片，使其居右，这样，页面中的文本就在图像的左侧排列。而第 20 行和第 22 行表示水平线，水平线的长度为 400px，宽度为 3px，令其左对齐。

4.6　小　　结

本章主要介绍了关于页面中图像的知识。重要的知识点有：

- 图像分为位图和矢量图，设计者经常将 JPEG、PNG 和 GIF 图像放在页面中。
- 图像的分辨率和像素。
- 学习了图像和图像之间排版的方法以及图像和文本之间排版的方法。
- 学习了使用 Windows 自带的画图工具来截取修饰图像的方法。
- HTML 中使用代码给图像添加边框和在页面中放入水平线。
- 设置页面背景并有效地控制页面背景。

最后，本章通过一个综合的实例展示了这些知识点的实际应用效果，这样的页面显然比纯文本的页面美观了一些，但是还不够。优秀的页面还需要更精致的装扮，如果页面背景不是白色的，会不会更美观。如果可以使用不同颜色的字体，会不会更绚丽。带着这样的疑问，将在第 5 章的知识中学习如何解决这些问题。

第 5 章　让网页变得更绚丽一些

页面设计中，难道所有的字体都应该是黑色的，所有的背景都应该是白色的，所有的图片都应该是方方正正的吗？当然不是。虽然代码语言规则是固定的，标签是永远不会改变的，但是在语言代码的基础上，设计者却可以花尽心思，开发出许多令人赞叹的、具备特殊视觉效果的页面。本章的知识点如下。

- 计算机的颜色原理。
- 如何使用颜色去修饰页面。
- 了解图像的通道。
- 图像的简单应用。

5.1　了解计算机语言下的颜色描述

在计算机的世界里，每种颜色都有自己的一串数字，如白色是#FFFFFF、黑色是#000000、红色是#FF0000、巧克力色是#D2691E 等，这串数字就是表示颜色的颜色值。在理解颜色值前，首先要明白计算机的颜色模式。

人类的眼睛看到的颜色有两种，一种是发光体发出的颜色，如电视机荧屏；另一种是物体本身不发光，而是反射光线产生的颜色。这就很常见了，生活中的大部分物质都是不发光的，如书本。而对于发光体的颜色模式，典型的应用便是计算机屏幕。计算机屏幕上的每一个点都有红、黄、蓝三色的荧光粉，在电子枪的照射下，发出不同比率的三原色光，相加混合形成用户所看到的图像。

这 3 种基本的颜色，称之为“三原色”，发光体的颜色模式，又称之为“加色模式”，两者相统一，便是 RGB 色彩模型。而网页的颜色模式，就是这种 RGB 模式来确定的。R、G、B 3 个颜色通道都使用 8 位存储器，这样每个颜色可以有 2 的 8 次方，也就是 256 个层次。所以很多软件中单个颜色通道都是用 0～255 的整数范围，共 256 个数来表示的，3 个颜色通道加在一起，这个色彩模型一共能表现 1670 万种颜色。按照这样的规律，以十六进制来表示，255 相对于十六进制下的 FF，然后把 3 个颜色值依次并列表示出来，并以#开头，如图 5.1 所示是一个经常适用于图形软件中的色板。

图中 R=255、G=255、B=255，即三色颜色值都为最大时，按照三原色理论，红色、绿色和蓝色加在一起时，得到的结果自然是白色了。同时，按照规律，3 个数字依次都是最大值，全部记作 FF，串在一起就是#FFFFFF。如果 3 个值都为最小值，颜色为#000000，既然是最小值，意味着没有任何颜色，这时，表现出来的当然就是黑色了。事实上，对于设计者来说，由于大部分软件都可以直观地选择颜色，所以并不需要刻意去背熟不同色彩的颜色值。



说明：颜色值的表示中，不区分大小写，所以#FFFFFF 和#ffffff 是一样的，都表示为白色。

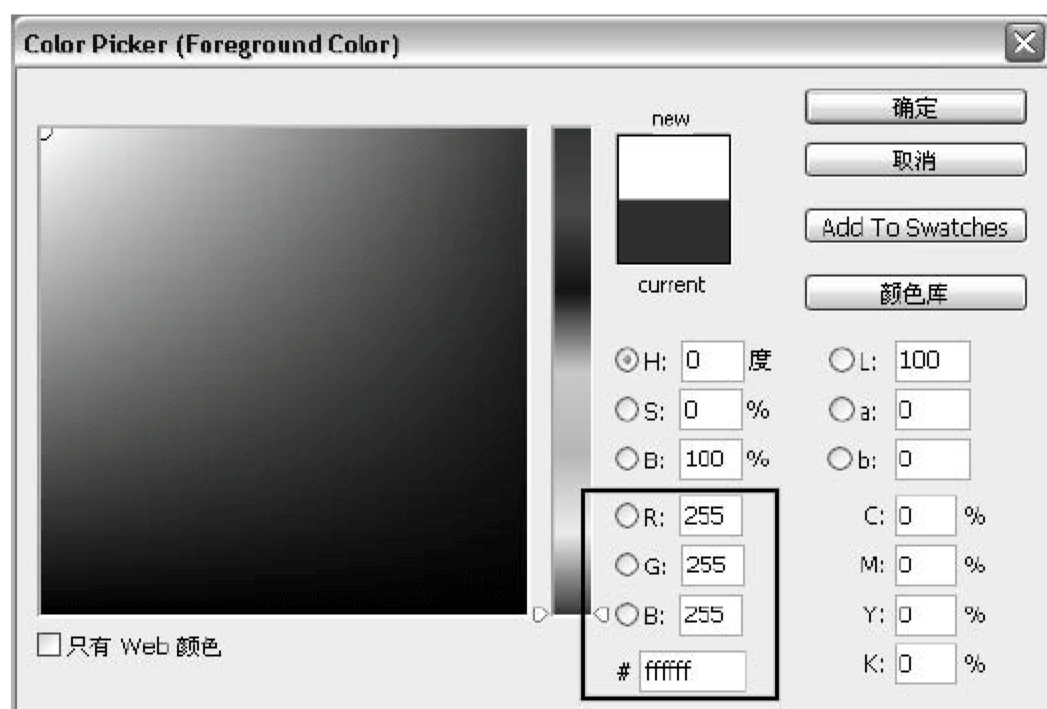


图 5.1　颜色值

5.2　让页面绚丽多彩

从第 4 章中了解到可以使用图像作为页面背景来装饰网页，使之美观。但如果只需要改变页面背景颜色，那是不是需要放入整整一大张单色的背景图片呢？当然是不需要的。HTML 标签提供了可以直接修改页面背景色的方法，甚至设计者可以使用标签修改文本字体的颜色，让页面更丰富。

5.2.1　改变页面背景颜色

HTML 标签的使用不难，类似于很多次已经学习到的样式，如背景图像，依然是在<body>标签中添加 style 属性。

```
<body style="background-color:…" >
```

在这句代码中，属性可以用颜色值表示，而且还支持使用标准的 Windows 颜色名词，有 Black、White、Red、Green、Blue、Yellow、Magenta、Cyan、Purple、Gray、Lime、Maroon、Navy、Olive、Silver、Teal。当然，对于了解如何调色的设计者来说，完全可以使用颜色值去尝试自己定义颜色，如#FF0000 是红色。如果设计者希望得到一种紫红色，那么便要在红色里面加上蓝色，所以数值表示就成了#FF00FF。如果希望用更直接的方法，可以直接在属性中输入颜色名词，程序 5.1 展示了颜色名词所对应的页面背景。

【本节示例参考：资料光盘\第 5 章\5-1　改变页面背景颜色.html】

【实例 5-1】改变页面背景颜色的方法，其源码展示如下：

程序 5.1　改变页面背景颜色.html

```
01  <html>
02    <head>
03      <title> 设置背景颜色</title>
04    </head>
05    <body style="background-color:teal">          //设置页面背景的颜色
06      <h5>
```

```
07          <br><br><br>背景颜色的设置，这是墨绿色</h5>
08      </body>
09    </html>
```

【运行程序】最终显示在浏览器中的结果如图 5.2 所示。



图 5.2　背景颜色的设置

5.2.2　改变页面文本字体颜色

修改页面文本的颜色，直接在结构性标签中添加颜色属性即可，代码如下：

```
<body style="color:…">
```

或者也可以写为：

```
<p style="color:…">
```

程序 5.2 展示了不同颜色名词下的文本颜色。

【本节示例参考：资料光盘\第 5 章\5-2　改变文本的颜色.html】

【实例 5-2】改变文本的颜色，其源码展示如下：

程序 5.2　改变文本的颜色.html

```
01    <html>
02       <head>
03         <title>文本颜色</title>
04       </head>
05       <body>
06    <p style="color:red">这是使用 Red 的文本
07    <p style="color:green">这是使用 Green 的文本
08    <p style="color:blue">这是使用 Blue 的文本
09    <p style="color:yellow">这是使用 Yellow 的文本
10    <p style="color:magenta">这是使用 Magenta 的文本
11    <p style="color:cyan">这是使用 Cyan 的文本
12    <p style="color:purple">这是使用 Purple 的文本
13    <p style="color:gray">这是使用 Gray 的文本
14    <p style="color:lime">这是使用 Lime 的文本
15    <p style="color:maroon">这是使用 Maroon 的文本
```

```
16  <p style="color:navy">这是使用 Navy 的文本
17  <p style="color:olive">这是使用 Olive 的文本
18  <p style="color:silver">这是使用 Silver 的文本
19  <p style="color:teal">这是使用 Teal 的文本
20      </body>
21  </html>
```

【运行程序】在浏览器中的显示结果如图 5.3 所示。



图 5.3　文本的颜色

【深入学习】这个页面展示了 Windows 标准颜色名词的显示效果，但从设计页面的角度来说，这里有个技巧，即一个页面中使用的颜色最好不要超过 4 种颜色，至于设计文本通常情况下就更不必用得五颜六色。颜色是一种语言，它更深层次的作用是传递页面的信息，如橙色、黄色显得活泼，红色显得有激情，绿色代表了健康祥和，蓝色代表了稳重、内敛、平静等。用错颜色有时也会表错意。

5.3　不寻常的图像应用

在页面中放入的图像文件大部分为 JPEG、GIF 和 PNG。JPEG 图像就如生活中看到的照片一样，没有什么特别之处，而计算机中的图像具有一些不同于照片的特性。如 GIF 图像和 PNG 图像就有一些特殊的效果，GIF 图像可以制作成简单的动画，而 PNG 图像带有透明通道，可以制作透明图像。

5.3.1　会动的 GIF 图像

GIF 图像可以用来制成逐帧动画，逐帧动画就是指将一幅幅图像在时间帧上依次绘制。如早年的动画片制作一样，动画和图像相比，其最大的优势是能够表现一个具体动作，传递给浏览者不一样的信息。GIF 动画也是占用空间最小的一种动画，所以在页面中得到广泛的使用，如程序 5.3 在页面中放入的是一个 GIF 动画。

【本节示例参考：资料光盘\第 5 章\5-3　使用 GIF 动画点缀页面.html】

【实例 5-3】使用 GIF 动画点缀页面的方法，其源码展示如下：

程序 5.3　使用GIF动画点缀页面.html

```
01          <html>
02            <head>
03              <title> 使用 GIF 动画文件</title>
04            </head>
05            <body>
06              <h3>会动的图像</h3>
07              <img src="图片/狗.gif">          //放入 GIF 动画
08             </body>
09           </html>
```

【运行程序】在浏览器中显示的结果如图 5.4 所示。



图 5.4　使用 GIF 动画

【深入学习】当在浏览器中查看这个页面时，宠物狗会眨眼睛，摆动它可爱的尾巴，这是因为放入的是 GIF 动画的图像，其本身和 HTML 放入图像的方法没有区别。页面中还有一种常用的动画文件——Flash。Flash 不仅能实现会动的图像，而且能制作出交互式的动画效果。Flash 也较 GIF 能制作更丰富的效果，可以表达时间更长的动画等。



说明：Flash 的内容将会在第 15 章中介绍。

5.3.2　图像的透明通道

了解图像的透明通道之前，首先要理解图层的概念，计算机中的图像并不像生活中的照片一样，拿在手中就是薄薄的一层纸。试想，如果把两张照片叠加在一起，当然，只能看到最上面的那张照片，但如果最上面的那张照片是透明的，人们就能完全看到下层照片的内容，如果那张照片是半透明的，那么下层的照片就可以若隐若现出来。在计算机中，设计者制作图像时，就是使用这样的一种思路来

制作图像。很多格式的图像都具有透明通道，如 PNG、BMP、TGA 等。如图 5.5 所示，这是一幅橙色背景的网页。图 5.6 分别是相同内容、不同格式的图像，图像格式分别为 JPEG 和 PNG。



图 5.5　橙色背景



图 5.6　cake 不同格式的对比

通过肉眼几乎很难分别 JPEG 图像和 PNG 图像的区别，如果仔细观察，可以发现 PNG 图像的背景色稍许偏灰色一些。接下来，将图 5.6 中的两幅不同格式的 cake 图像放入到图 5.5 的橙色背景页面中去，如程序 5.4 所示。

【本节示例参考：资料光盘\第 5 章\5-4　对比 JPEG 图像和 PNG 图像.html】

【实例 5-4】对比 JPEG 图像和 PNG 图像，其源码展示如下：

程序 5.4　对比JPEG图像和PNG图像.html

```
01        <html>
02          <head>
03            <title>对比 PNG 和 JPEG 图像</title>
04          </head>
05          <body style="background-color:ffcc00">
06            <img src="图片/cake.jpg">
07            <img src="图片/cake.png">        //分别放入不同格式的图像
08               </body>
09          </html>
```

【运行程序】最终在浏览器中显示的结果如图 5.7 所示。



图 5.7　对比 JPEG 图像和 PNG 图像



注意：本小节使用的是 Firefox 浏览器，是因为 IE 6.0 之前版本的浏览器存在并不能直接显示 PNG 图像的问题。现在大部分 Windows 操作系统下自带的都是 IE 7.0 版本以上浏览器。

【深入学习】通过比较，发现 JPEG 图像的背景依然是白色，而 PNG 图像的背景已经消失了，露出了网页背景的颜色，这就是透明通道作用下的效果。利用 PNG 图像的这种特性，页面设计中的图像便不再局限于看上去是个矩形，图像显得自然生动。

那究竟什么是透明通道？从原理上来说，颜色由基本三原色构成，所以计算机中图像具备 3 个基本通道，红色通道、绿色通道和蓝色通道。图像如同是由三原色的图层叠加而成，而此外，在此基础上再加上一个透明通道来控制图像的透明度。透明通道又称之为 Alpha 通道，是一个 8 位的灰度通道，该通道用 256 级灰度来记录图像中的透明度信息。所以，PNG 图像是带有透明通道的，而 JPEG 图像不带有透明通道。

5.3.3 带有透明通道图像的应用

正因为 PNG 图像可以保留透明通道，图像放在页面中不会出现白色边缘，如果在已有背景颜色的页面上再添加背景图像，就可以做好图像和背景色很好的融合。这样对于设计者来说，可以作为背景素材的选择就大大增加了。

此外，由于透明通道可以控制图像的透明度，当图像设置一定数值的透明度时，可以使背景图像若隐若现地表现出来，如阴影效果。如图 5.8 所示是一张画面内容为圣诞树的图像，这里分别使用 JPEG 和 PNG 格式来制作页面背景。程序 5.5 将这张图像设置为页面背景。

【本节示例参考：资料光盘\第 5 章\5-5　同时使用背景图像和背景颜色的效果.html】

【实例 5-5】同时使用背景图像和背景颜色的效果，其源码展示如下：

程序 5.5　同时使用背景图像和背景颜色的效果.html

```
01        <html>
02          <head>
03           <title>同时使用背景图像和背景颜色的效果</title>
04          </head>
05          <body bgcolor="#99CC33"                    //设置页面背景颜色
06                background="图片/圣诞树.png">          //设置页面背景图像
07          </body>
08        </html>
```

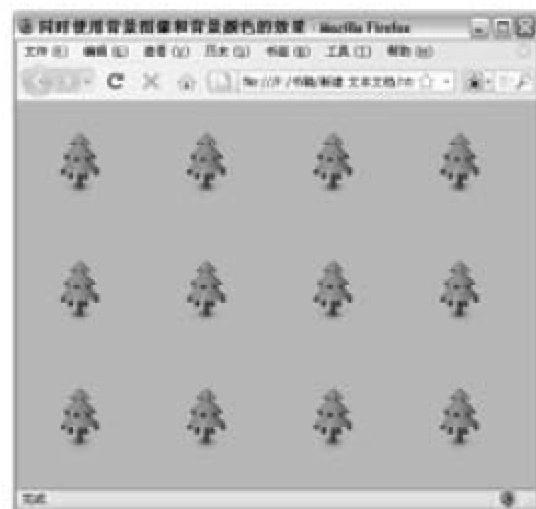


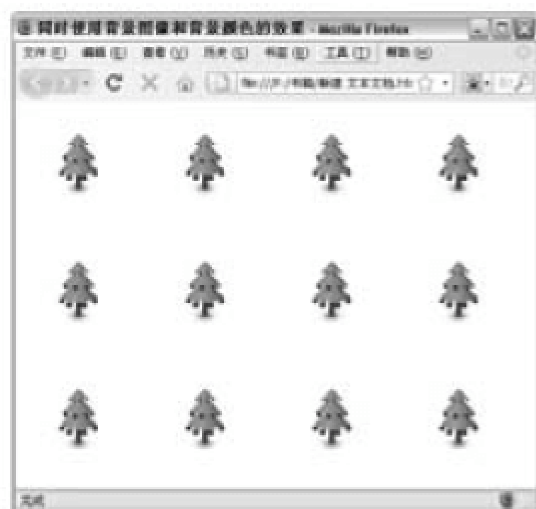
说明：实例 5-5 中使用的是 PNG 图像，如果使用 JPEG 图像，更改图片格式就可以了。

【运行程序】这个页面最终的浏览效果如图 5.9 所示。



图 5.8　圣诞树





PNG图像页面背景　　JPEG图像页面背景

图 5.9　不同格式图像配合背景颜色的效果

【深入学习】代码第 5 行中“#99CC33”，说明使用了绿色的背景颜色。而从图 5.9 中对比可以发现，JPEG 图像作为背景图像时，带有白色背景的“圣诞树”图像会覆盖后面的背景色。

也许去除图像的白色背景并不是十分的有意思，所以 PNG 图像更进一步的使用将更神奇。阴影效果是其中最典型、最常见的一种用法。在页面中，为了突出文字标题，如导航栏、目录栏等。通常设计者会将文字转换成 PNG 图像，应用不同的效果，如添加阴影、发光边缘，使之成为页面的焦点。图 5.10 是已经做好阴影的图像。

图中“阴影效果”图层背景的棋盘格是 Photoshop 中表示空白图层的特定使用。所以，图像中的内容实际上只有 4 个字和字体所带有的阴影部分。而阴影在图像中是一种半透明状态，意味着可以透过阴影图像略略看到后面的图层，在页面中，后面的图层就是页面的背景色。程序 5.6 将这张图像放入页面中。



说明：空白图层是指没有任何内容，包括白色背景的图层。

【本节示例参考：资料光盘\第 5 章\5-6　阴影效果配合页面背景色.html】

【实例 5-6】阴影效果配合页面背景色，其源码展示如下：

程序 5.6　阴影效果配合页面背景色.html

```
01        <html>
02          <head>
03            <title>阴影效果配合页面背景色</title>
04          </head>
05          <body body bgcolor="#99CC33">          //设置好页面的背景色
06            <img src="图片/阴影效果.png">          //放入图像
07          </body>
08        </html>
```

【运行程序】最终在页面中的显示效果如图 5.11 所示。



图 5.10　阴影效果



图 5.11　页面中阴影效果

【深入学习】在页面中，配合背景色或背景图像时，漂亮的效果就被呈现出来。所以在页面中，有些文本的内容，其中相当一部分都是以图像的形式展示出来，利用 PNG 图像的透明通道来添加这样的效果。当然，阴影效果只是其中一种，有兴趣的读者可以充分发挥创作想象，在页面中实现更有趣的效果。

5.4　案例：修饰普通页面

在第 4 章的 4.5 节中制作了一个普通的描述宠物的页面。在本章中，将实践如何将这个普通页面修饰得更加漂亮。在这个例子中，首先将给宠物页面改动页面背景，同时为了进一步美观，将“Jenny”文本标题转换成 PNG 图像，这样设计者可以赋予标题更多的设计，如程序 5.7 将这些设计的想法转化成代码。

【本节示例参考：资料光盘\第 5 章\5-7　进一步修饰宠物页面.html】

【实例 5-7】进一步修饰宠物页面，其源码展示如下：

程序 5.7　进一步修饰宠物页面.html

```
01  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03    <html xmlns="http://www.w3.org/1999/xhtml">
04          <head>
05              <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
06              <title>进一步修饰宠物页面</title>
07        </head>
08        <body background="图片/宠物页面背景.png">
09              <!—设置页面背景 -->
10            <p><img src="图片/我的宠物 banner.gif" width="1017" height="24" /></p>
11              <!—插入页面 banner -->
12            <p><img src="图片/jenny.png">
13              <!—插入页面图像 -->
14            <p><img src="图片/我的宠物.gif" width="192" height="262" border="2" align="right"
15            /></p>
```

```
16              <!—插入页面图像 -->
17              <p>年龄：24<br/>
18                  最喜爱的食物：小虾、肉<br/>
19                  特点：很安静</p>
20              <p><hr align="left" width="400" size="3" >
21              <p> <h3 style="color:red">  更新宠物日记</h3>
22              <hr align="left" width="400" size="3" >
23              <p>2008 年 9 月 1 号</p>
24              <p>  外面的天气还是很热，所以 Jenny 一直都懒懒的趴在那里，好久也<br>不愿意动一下，
25          当然，看到好吃的食物时还是会激动那么一下的，<br>Jenny 很聪明，看到主人来了，就会探出脑袋，
26          一直盯着你。 <br />
27          </body>
28 </html>
```

【运行程序】浏览该页面，运行的结果如图 5.12 所示。

【深入学习】代码第 8 行中添加了背景图像，第 10 行将原先的 Jenny 文本转化成 PNG 图像，由此添加了更多的效果，如文字描边、图像阴影，在页面中显得更加醒目。在第 21 行中，将文本修饰为红色，以达到提高吸引眼球的作用。虽然是小小的改动，但在页面中显示出来却是大大的改观，如图 5.12 所示是这个页面的最终效果。



图 5.12　进一步修饰宠物页面

5.5　小　　结

本章的知识是基于第 4 章图像的运用之上，计算机世界中的图像是有别于现实图像的。本章的内容主要就是针对于这些不同之处，如何在页面中充分发挥计算机图像的作用。本章给出了一些重要的图像基础知识，主要的知识点有：

- 计算机中颜色数量有限，由每一个颜色值来描述每一种颜色。
- 改变页面背景色。
- 修改文本颜色。
- GIF 图像的动画效果。
- 用 PNG 图像的透明通道来展示阴影效果。

在第 6 章中将介绍网页的链接，网页之所以能够成为网站，就是因为有许多的页面链接在一起，这也是网页独具魅力的地方。

第 6 章　网 页 链 接

一个普通的网站，就是将许多页面链接在一起，用户通过网站的主页查找网站中的所有页面，而网页彼此之间的链接，则称之为页面的链接。这就像在书本中查找目录时，在第一页的目录中找到所需资料的所在书页，然后根据所在页数翻找到此页。而在网站中，用户在页面中选择链接内容，页面则会自动跳转到所在的页面。本章的主要内容如下。

- ❑ 理解网页链接。
- ❑ 学习基本的链接形式。
- ❑ 如何修饰链接的状态。
- ❑ 了解特殊形式的链接。

6.1　网页链接概述

所谓的链接是指从一个页面指向一个目标的链接关系，这个目标是多种样式的，可以是一个网页，也可以是相同网页的不同位置，甚至可以是一张图片、一个电子邮件地址、一个应用程序。当用户单击已经链接的页面内容时，链接目标将显示在浏览器上，并根据目标的类型来运行。

可以说，在一个大型网站中，网页的链接已经充斥着网站中的每个角落。如图 6.1 所示是一家门户网站的主页。在这样的一个主页中，用户可以看到的每一行文本、每一张图片，几乎所有的页面内容都是页面链接。

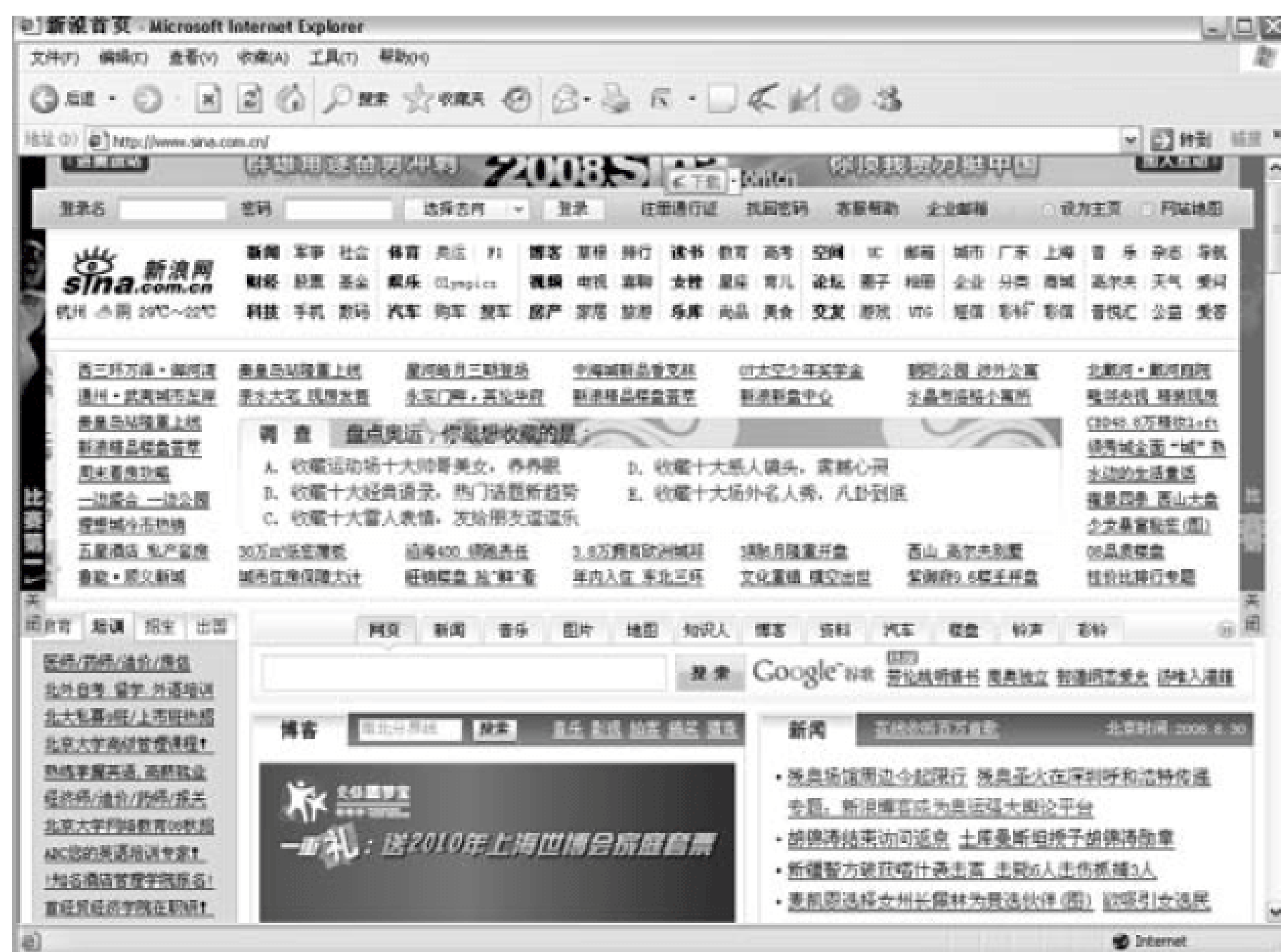


图 6.1　链接主页

6.1.1　初识页面链接

在 HTML 文档中，使用<a>标签指引页面中链接的目标点，让设计者创建指向目标点的链接。在链接的属性中，代码的写法为：

```
<a href="链接对象的路径">链接锚点对象
</a>
```

用 a 来表示锚点，a 也源自于英文中的 anchor。href 属性的意思是超文本引用，这个属性的值指定了链接的目标。在一个完整的链接语句中包含两个部分，即链接锚点对象和链接地址，如程序 6.1 是一个简单的链接页面，程序 6.2 是另一个页面。通过链接的方法，将使程序 6.1 的 A 页面跳转到程序 6.2 的 B 页面。

【本节示例参考：资料光盘\第 6 章\A.html　B.html】

【实例 6-1】页面 A，其源码展示如下：

程序 6.1　A.html

```
01        <html>
02          <head>
03            <title>页面 A </title>
04          </head>
05          <body>
06            <h1><a href="B.html">A</a> </h1>            //超链接到页面 B
07            <h1>B</h1>
08          </body>
09        </html>
```

【实例 6-2】页面 B，其源码展示如下：

程序 6.2　B.html

```
01        <html>
02          <head>
03            <title>页面 A </title>
04          </head>
05          <body>
06         <h1>页面 A </h1>
07         <h1><a href="A.html">页面 B</a></h1>         //超链接到页面 A
08          </body>
09        </html>
```

【运行程序】最终在浏览器中的结果如图 6.2 所示。

【深入学习】单击“页面 A”链接时，页面会跳转到 B 页面，单击“页面 B”链接，也会跳转回 A 页面。在 A、B 两个页面代码中，页面 A 代码中的第 6 行和页面 B 代码中的第 7 行，放入<a>标签中的内容便是链接的锚点对象，如文本“页面 A”和“页面 B”。而在<a>标签中，href 属性下的内容

如 A.html 和 B.html，这两个页面存放的地址即是链接的地址，这两个元素结合在一起，便完成了一个链接过程。



图 6.2 理解页面链接

6.1.2 理解链接地址

链接地址指的是链接到锚点对象的路径，这个路径所指的不仅是一个页面地址，也可能是一个文件地址、一个邮箱地址。那么，对于一个页面的链接，该如何去定位这个链接对象的路径呢？如程序 6.3 是通过一个链接地址跳转到另一个页面，结果如图 6.3 所示。

【本节示例参考：资料光盘\第 6 章\6-3 链接内容的路径.html】

【实例 6-3】链接内容的路径，其源码展示如下：

程序 6.3 链接内容的路径.html

```
01        <html>
02          <head>
03            <title>链接内容的路径</title>
04          </head>
05          <body>
06         <h1>链接内容的路径</h1>
07         <h1><a href="图片/时间表.jpg ">时间表             //图像即是链接的目标
08               </a></h1>
09          </body>
10        </html>
```

【运行程序】浏览该页面，运行的结果如图 6.3 所示。



图 6.3 链接内容的路径

【深入学习】单击“时间表”链接时，页面即跳转到图片，而图片则显示在新的页面中。在这个例子中，从图 6.3 中的浏览器地址栏可以看出，以.html 为后缀的页面文件“6-3 链接内容的路径.html”是放在命名为“资料光盘”的文件夹下，而页面链接的内容，即 JPEG 文件“时间表.jpg”，是放在“图片”文件夹中，而“图片”文件夹又是放在“资料光盘”文件下，因而页面文件和“图片”文件夹属于同一目录下。这样层层递推的关系可以令浏览器找到这张图像，如图 6.4 中文件夹的位置。



图 6.4　页面文件和图片文件夹



说明：资料光盘中页面图像的位置：资料光盘\第 6 章\图片文件夹。

所以在代码中定义链接对象的路径时，有这样一个规律：所链接的内容，如实例 6-3 中“时间表.jpg”，从和页面文件同一目录下的文件夹起开始定义，如实例 6-3 中“6-3 链接内容的路径.html”是页面文件。而“图片”文件夹是和页面文件属于同一目录下。为了正确引用图像的路径，代码第 7 行中定义图像的路径，就需要从“图片”文件夹这个位置开始定义。



注意：如果将页面文件“6-3 链接内容的路径.html”放在文件夹 F 盘中，那么链接地址的路径就应该改为“<a href="资料光盘/图片/时间表.jpg">”。

6.2　链接的种种不同

链接的分类有种种不同，使用的方法却大同小异，创建一个超链接很容易。事实上，设计者使用到的只有<a>标签而已。虽然链接的方法类似，但其展示的形式却自由多变，如链接的方式、链接指向何处等。而从使用者的角度来说，设计者最重要的是保持链接的友好性。

6.2.1　基本的文本链接

文本的链接是页面中最常见的链接形式，也是最基本的一种链接使用。一般文本链接中，最初文字上的超链接呈蓝色，文字下面有一条下划线，如果超链接已经被浏览过了，文本颜色就会发生改变，默认是紫色。设置文本的链接时，在文本的段落中直接使用<a>标签。如程序 6.4 是一个文本的链接。

【本节示例参考：资料光盘\第 6 章\6-4　文本链接.html】

【实例 6-4】文本链接的代码如下：

程序 6.4　文本链接.html

```
01          <html>
02            <head>
03              <title>蝙蝠侠之黑暗骑士</title>
04            </head>
05            <body>
06              <h3>影片《蝙蝠侠之黑暗骑士》</h3>
07             可怕的黑暗渐渐散去的哥谭市，逐渐恢复了往日的平静。犯罪率呈直线下降……
08             凶残的小丑带着他                        <!--“小丑”是链接的位置-->
09            </body>
10          </html>
```

【运行程序】结果如图 6.5 所示，将链接的锚点对象放入<a href>标签内就可以完成链接了，当单击文本“小丑”链接时，页面跳转到链接的页面。



图 6.5　文本链接

6.2.2　基本的图像链接

图像链接的使用频率和文本链接一样相当高，设置链接的方法和文本无异，在引用图片的代码前面先放入<a>标签。代码如下：

```
<a href="…">
<img src="…">
</a>
```

【本节示例参考：资料光盘\第 6 章\6-5　设置图像链接.html】

【实例 6-5】设置图像链接的方法，其源码展示如下：

程序 6.5　设置图像链接.html

```
01          <html>
02            <head>
```

```
03              <title>图像的链接</title>
04            </head>
05            <body>
06              <h3>图像的链接</h3>
07                <a href="苏轼简介.html">                //这里链接到目标页面
08                <img src="图片/吏部陈公诗跋.jpg">    //这张图像是链接的位置
09               </a>
10            </body>
11          </html>
```

【运行程序】浏览效果如图 6.6 所示。



图 6.6　图像的链接

【深入学习】代码中第 6～8 行即是对页面文档调用的图像设置链接。依照这种方法，对于设置 Flash 文件或是引用其他文件的链接都是一样的。

此外，有时使用的图片是非正规矩形时，图像会出现边框，如图 6.6 中的图像边框。如果想去除边框，可以在代码中添加代码“border=0”。那么代码第 8 行则可以写为“<img src="图片/吏部陈公诗跋.jpg" border=0>”。

> **说明：使用 CSS 样式表修饰图像会是一种更好的方法。**

6.2.3　把邮箱留给需要联系你的人

<a>标签不仅是网页和网页之间，或者网页和文件的链接，还可以链接电子邮箱地址。这是通过网页让使用者和设计者联系的最方便的方法。当然，也可以直接在页面中留下电子邮件地址，但有时为了突出友好性，不是直接留下地址，而是采用将邮箱链接到页面内容上的方式，使用方法如下：

```
<a href="mailto:邮箱地址">链接锚点对象
</a>
```

mailto 其实就是 mail to 的连写，意思是“把邮件发送到”。在这行代码中，还可以给新邮件填好

邮件的主题和正文，这样打开电子邮件程序就填好了收信人的新邮件。这通过属性 subject 和 body 来实现，使用时有些特别，需要放在两个问号之间，如程序 6.6 链接到的邮箱地址。

【本节示例参考：资料光盘\第 6 章\6-6　链接邮箱地址.html】

【实例 6-6】链接邮箱地址的方法，其源码展示如下：

程序 6.6　链接邮箱地址.html

```
01        <html>
02          <head>
03            <title>邮箱的链接</title>
04          </head>
05          <body>
06            <h1>邮箱链接</h1>
07             <a href="mailto:huizhao@foxmail.com?subject=联系我?body=告诉我们你对网页设
08             计的想法?"> 告诉我们你对网页设计的想法?
09          </body>
10        </html>
```

邮箱地址

【运行程序】在浏览器中的效果如图 6.7 所示，当单击“告诉我们你对网页设计的想法”链接时，系统会自动打开邮件客户端。



图 6.7　邮箱的链接

注意：如果浏览者的系统没有安装邮件客户端，则会给使用者带来很大麻烦。因此，这种使用方法在某些时候也遭到设计者的排斥。

【深入学习】互联网中充斥着很多垃圾邮件的制造者，他们创建电子邮件的数据库，然后给电子邮件地址发送大量的垃圾邮件。这其中有一种方法就是使用程序自动搜索含有 mailto 的链接。所以为了防止垃圾邮件，有时也把邮件地址的英文字母转换成 ASCII 字符。例如，在实例 6-6 的邮箱地址 huizhao@foxmail.com 中改动其中一个或部分字母，在 ASCII 字符中小写字母“a”的代码是“a”，那么邮件的地址可写成“huizhao@ foxmail.com”，这样程序搜索出来的便是乱码的电子邮件地址。

6.2.4　在同一页面中快速查找信息

页面除了和页面之外的文件或程序链接外，也可以和同一页面中的内容进行链接。这种情况通常用于导航，为的是浏览页面的人直接可以跳到自己需要的信息版块上。由于是在同一页面内实现链接，也就是说，页面链接的路径就是在同一页面内，所以在 HTML 语言中使用<a>标签中的 id 属性来确定路径位置。通过以下两个步骤可以理解这种代码的用法。

（1）要确定链接的锚点对象，不同于页面和外部文件链接的方式，链接的路径由于在同一页面内，这里需要使用“#”来引用同一页面中的内容。代码如下：

```
<a href=#...>
<a>
```

（2）需要在页面中设定出链接的目标。使用的就是 id 属性。

```
<a id=…>
```



说明：id 也可以写成 name，区别在于 name 是 HTML 的标准，而 id 是 XHTML 中的标准要求。

id 属性后放入的内容，就是第（1）步中 href 属性下设定好的内容。这样前后呼应，自然就很容易找到位置。

【本节示例参考：资料光盘\第 6 章\6-7　同一页面内实现链接.html】

【实例 6-7】同一页面内实现链接，其源码展示如下：

程序 6.7　同一页面内实现链接.html

```
01          <html>
02            <head>
03              <title>同一页面内实现链接</title>
04            </head>
05            <body>
06              <h2>世界博览会</h2>
07                <a href=#概述>概述</a>
08            <br><a href=#世界博览会的历史与由来>世界博览会的历史与由来</a>
09       <!--设置页面的锚点链接 -->
10            <br><a href=#国际博览局与世界博览会>国际博览局与世界博览会</a>
11       <!--设置页面的锚点链接 -->
12            <br><a href=#中国与世界博览会>中国与世界博览会</a>
13       <!--设置页面的锚点链接 -->
14    <p>
15          <h3><a id=概述></a>概述</h3>
16              <!--锚点的位置 -->
17              <br>  世界博览会（World Exhibition or Exposition，简称 World Expo）是一项由主办
```

```
18          国政府组织或政府委托有关部门举办的有较大影响和悠久历史的国际性博览活动。它
19          ……
20              <br>  世界展览会的会场不单是展示技术和商品，而且伴以异彩纷呈的表演，富有
21          魅力的壮观景色，设置成日常生活中无法体验的、充满节日气氛的空间，
22          ……
23          <p><h3><a id=世界博览会的历史与由来></a>世界博览会的历史与由来</h3>
24              <!--锚点的位置 -->
25              <br>  在古代农耕社会，人们往往在庆贺丰收、宗教仪式、欢度喜庆的节日里展开
26          交易活动，后来逐渐发展成为定期的、有固定场所的、以物品交换为目的的大型贸易
27            ……
28              <p><h3><a id=国际博览局与世界博览会></a>国际博览局与世界博览会</h3>
29          <!--锚点的位置 -->
30              <br>  举办世界博览会的目的往往是为庆祝一个重大的历史事件或一个地区、一个
31          国家的重要纪念活动：为了展示人类在单一或多个领域中，政治、经济、文化、科技等方面
32          ……
33              <p><h3><a id=中国与世界博览会></a>中国与世界博览会</h3>
34          <!--锚点的位置 -->
35              <br>  中国第一次参加世界博览会应该始于 1873 年的维也纳世界博览会，中国的
36          参加方式也是赛奇会本身的一奇。因为代表中国的是一个叫包腊（EoCoBowra）的英国
37          ……
38        </body>
39    </html>
```

【运行程序】浏览该页面，结果如图 6.8 所示。



图 6.8　同一页面内的链接

【深入学习】由于页面太长，当单击目录栏时，页面会跳到相应的目标位置，以方便浏览者阅读页面信息，这是提高页面友好性的一个很好的方法。在这个例子中，注意第 8、10、12 行和第 23、28、33 行之间的呼应。

在图中单击“世界博览会的历史与由来”目录时，页面自动跳转到“世界博览会的历史与由来”内容版块。从代码上看，第 23 行定义了一个 id 属性，如同在页面中声明“这里是一个目标位置”，而在第 8 行代码中设置了超链接，表明“链接到这个页面第 23 行的位置”。

6.3　提高页面链接的友好度

在设置了超链接的文本中，链接的内容都带有下划线，浏览过的字体也都是特定的颜色，始终给人千篇一律的感觉，而对于浏览者来说，这是一种不太舒服的感受。为了解决这些问题，使页面展现出亲和力的一面，设计者总是会用一些新颖的方法去改变链接的状态。

6.3.1　美观链接的状态

链接的状态在页面中是很显眼的一部分，起到的作用举足轻重，而链接的样式是可以通过定义来修改的。在修改之前，首先要搞明白链接的过程，一个链接状态，可以分解为以下 4 个步骤：

（1）链接还未被访问。

（2）链接被选中时。

（3）鼠标滑过链接。

（4）链接被访问后。

使用 HTML 标签属性，通过添加 link、alink 和 vlink 来修改超链接文本的颜色。link 属性修改链接未访问时的文本颜色，alink 属性修改链接被选中时文本的颜色，vlink 属性修改链接被访问后的文本颜色。程序 6.8 使用标签来修改链接的文本颜色。

【本节示例参考：资料光盘\第 6 章\6-8　使用标签属性修改文本链接颜色.html】

【实例 6-8】使用标签属性修改文本链接颜色的方法，其源码展示如下：

程序 6.8　使用标签属性修改文本链接颜色.html

```
01	<html>
02	  <head>
03	    <title>使用标签属性修改文本链接颜色</title>
04	  </head>
05	  <body  link=teal alink=red vlink=silver>
06	    <h3>使用标签属性修改文本链接颜色</h3>
07	    <a href="后退.html">注意文本颜色前后变化</a>
08	  </body>
09	</html>
```

【运行程序】浏览该页面，结果如图 6.9 所示，链接的文本前后的颜色发生了改变。



说明：当页面被访问过后，再次打开页面时，会发现链接始终保持访问后的状态，这是因为浏览器记录了用户的链接记录。如果想删除记录信息，可以在浏览器中选择“工具”|“Internet 选项”命令，在弹出的“Internet 选项”对话框中单击“清除历史记录”按钮即可清除记录信息。



图 6.9　使用标签属性修改文本链接颜色

【深入学习】这里是使用 HTML 标签属性来实现的功能，事实上这种旧方法并不值得推荐，更好的方法是使用 CSS。除了结构性的标签无法替代，如<body>、<p>，在表现性的作用上，应该习惯于避免使用标签属性的用法。而且 CSS 可以包含更多的属性修改，实现自由度更大的修饰。接下来从 CSS 的角度来了解如何修改链接状态。

- 链接还未被访问。

```
a:link {…}
```

- 链接被选中时。

```
a:active {…}
```

- 鼠标滑过链接。

```
a:hover {…}
```

- 链接被访问后。

```
a:visited {…}
```

在{}中通常添加两个基本的属性：color 属性修改文本的颜色，text-decoration 选择是否显示下划线。比较常见的用法是，设置未访问前的状态、设置被访问后的状态和设置划过鼠标链接的状态。所以，一个基本的使用 CSS 样式表如程序 6.9 所示。

【本节示例参考：资料光盘\第 6 章\6-9　使用 CSS 属性修改文本链接颜色.html】

【实例 6-9】使用 CSS 属性修改文本链接颜色，其源码展示如下：

程序 6.9　使用CSS属性修改文本链接颜色.html

```
01          <html>
02            <head>
03               <title>使用 CSS 属性修改文本链接颜色</title>
04              <style type=text/css>
05                      a {color:teal;
06                         text-decoration:none       //链接的状态，去除链接的下划线
07                        }
08                      a:visited {color:silver;
```

```
09                    text-decoration:none         //被访问后的链接状态
10                    }
11                  a:hover {color:red;
12                    text-decoration:underline    //划过链接文本的样式
13                    }
14            </style>
15          </head>
16          <body>
17            <h3>使用 CSS 属性修改文本链接颜色</h3>
18            <a href="后退.html">注意文本颜色前后变化</a>
19          </body>
20        </html>
```

【运行程序】浏览该页面，其效果如图 6.10 所示，单击前的状态是墨绿色，被访问之后就成了银灰色，而当鼠标划过链接文本时，显示的状态是红色带有下划线的文本。



图 6.10　使用 CSS 属性修改文本链接颜色

6.3.2　奇妙特殊的链接方式

6.3.1 小节中了解了通过使用 CSS 的方法可以去除链接默认的下划线。本节中将介绍两种新的方法来改变下划线的样式。首先需要了解两个属性：border-bottom 属性和 padding-bottom 属性。前者的意思是底部边界，后者的意思是底部内边。顾名思义，它们都是用来描述边框性质的属性。那么，这里的原理就是使用边框属性来替换原来的下划线。如程序 6.10 展示了这两个属性的作用。

【本节示例参考：资料光盘\第 6 章\6-10　使用 border-bottom 属性替换链接下划线.html】

【实例 6-10】使用 border-bottom 属性替换链接下划线，其源码展示如下：

程序 6.10　使用border-bottom属性替换链接下划线.html

```
01        <html>
02          <head>
03            <title>特殊的链接方式</title>
04            <style type=text/css>
05              a {
```

```
06                text-decoration: none;
07                border-bottom: 5px dotted red;      //改变下划线的样式
08                  }
09              </style>
10          </head>
11          <body>
12              <h3>点状的下划线
13              <p><h3><a href="后退.html">使用“border-bottom”属性替换链接下划线</a>
14          </body>
15      </html>
```

【运行程序】如代码中第 7 行的意思是“大小是 5px 的点状红色底部边框”。所以不难发现，这里是通过大小、形状和颜色来控制边框的形态，如图 6.11 所示。



图 6.11　点状的下划线

【深入学习】这里的 dotted 是点状的意思，除此之外，CSS 中还允许其他形状的下划线。其他的还有 dashed（虚线）、double（双线）、groove（槽线）、ridge（脊线）、inset（内陷）、outset（外陷），有兴趣的读者可以尝试一下。

padding-bottom 属性的作用可以引用自定义图像来制定下划线，技巧在于要排版好下划线和文本的距离，如程序 6.11 设计的自定义下划线。

【本节示例参考：资料光盘\第 6 章\6-11　使用 padding-bottom 属性替换链接下划线.html】

【实例 6-11】使用 padding-bottom 属性替换链接下划线，其源码展示如下：

程序 6.11　使用“padding-bottom”属性替换链接下划线.html

```
01      <html>
02          <head>
03              <title>特殊的链接方式</title>
04              <style type=text/css>
05               a {
06                text-decoration: none;                       //去除下划线
07                padding-bottom: 15px;                       //设置底边边界的位置
08                background: url(图片/手.png) bottom repeat-x;      //替换为自定义的图像
09                  }
10              </style>
11          </head>
12          <body>
```

```
13                <h3>使用自定义图像的下划线
14                <p><h3><a href="后退.html">使用“padding-bottom”属性替换链接下划线</a>
15            </body>
16        </html>
```

【运行程序】浏览该页面，在浏览器中的结果如图 6.12 所示。



图 6.12　使用 padding-bottom 属性替换链接下划线



注意：需要用心不断调整图像的位置才能设置好下划线，但是在不同的浏览器中可能会产生不同的结果，所以最好不要这样设置下划线。

【深入学习】代码第 7 行中控制了自定义的下划线和文本的距离，这个距离需要仔细控制，如果这个距离控制不当，在页面中会显示不完整图像或者显示不出自定义下划线。代码第 8 行的意思是“图像在 x 方向上重复的内边框”，而使用的这个图像是“手.png”。当然，在正规的网页设计中，最好不要这么做，因为这样会让使用者觉得很不自在。

6.3.3　热点图像区域的链接

所谓图像热点区域，就是指一个图像中的某一区域，那么热点图像区域的链接，自然就是使用这个区域作为超链接，就好像在一张地图上，以其中某一区域作为超链接。所以，在代码中也用到一个形象的标签——<map>标签。<map>标签下，嵌入使用<area>标签表明某一区域，其中有 3 个属性值来确定这个区域，分别是 shape 属性、coords 属性和 href 属性。

- ❑ shape 属性：用来确定选区的形状，分别是 rect（矩形）、circle（圆形）和 poly（多边形）。
- ❑ coords 属性：用来控制形状的位置，通过坐标来找到这个位置。一般来说，在实际操作中，设计者都会选择借助可视化的编辑页面的软件来实现这一功能，这就省却了花费很多心思在图像上测算具体的坐标值。
- ❑ href 属性：就是超链接。

所以，将这些属性运用在一起，这种方法的具体代码如下：

```
<map id=…>
    <area shape="…" coords="…" href="…">
</map>
```

这里介绍一种借助 Dreamweaver 软件来制作热点图像链接的实例。Dreamweaver 的工作界面如图 6.13 所示。

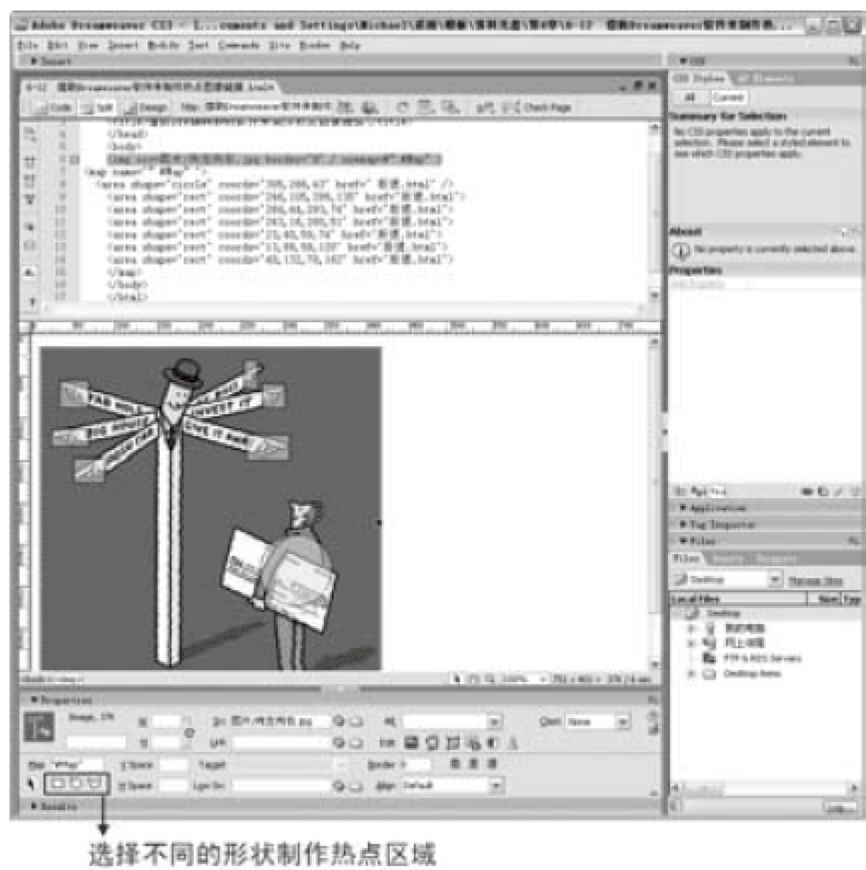


图 6.13　Dreamweaver 中制作热点区域

在 Dreamweaver 标准工作界面中，上部分是代码区，可以在这里写代码，中间是预览页面的地方，最下面是修改一些属性值的面板，右侧是一系列不同的工作面板。在这里并不需要使用到。当使用代码在页面中置入图像以后，在图 6.13 中的左下角单击线框中的图形按钮，Dreamweaver 中便直观地表示了不同形状热点区域的图标。选中后，在预览页面区域中的图像上绘制需要的形状并放置在需要的位置。设置好以后，代码区域会自动生成<map>标签，这里要修改两个属性。这时在代码区域中可以看到如下代码：

```
01<img src=图片/向左向右.jpg / usemap="#Map">
02    <map id="Map">
03       <area shape="circle" coords="303,265,86" href="#" />
04    </map>
```

在这个默认的代码中，第 2 行中 id 属性下为 Map，这个名字可以自行去定义。注意在第 1 行中，引用了这个命名为 Map 的热点区域链接。而在第 3 行的<area>标签中，shape 和 coords 属性已经自动生成。在这个例子中，表示为圆形的选区，位置定义在“303，265”的坐标位置上，尾数 86 代表的是这个圆的半径值，这个数值控制圆面积的大小。完整的页面源码如程序 6.12 所示。

【本节示例参考：资料光盘\第 6 章\6-12　借助 Dreamweaver 软件来制作热点图像链接.html】

【实例 6-12】借助 Dreamweaver 软件来制作热点图像链接，其源码展示如下：

程序 6.12　借助Dreamweaver软件来制作热点图像链接.html

```
01            <html>
02              <head>
03                <title>借助 Dreamweaver 软件来制作热点图像链接</title>
04              </head>
05              <body>
06            <img src=图片/向左向右.jpg border="0" / usemap=#Map>
07             <map name="Map">
08                <area shape="circle" coords="305,266,43" href="后退.html" />
```

```
09              <area shape="rect" coords="246,105,298,135" href="后退.html">
10              <area shape="rect" coords="264,44,293,74" href="后退.html">
11              <area shape="rect" coords="243,16,260,51" href="后退.html">
12              <area shape="rect" coords="23,40,59,74" href="后退.html">
13              <area shape="rect" coords="13,98,59,120" href="后退.html">
14              <area shape="rect" coords="40,132,78,162" href="后退.html">
15            </map>
16          </body>
17      </html>
```

【运行程序】这个页面在浏览器中的结果如图 6.14 所示。



图 6.14　热点区域链接

6.4　在新窗口中显示链接窗口

在先前的所有链接中，页面都是在同一页面中跳转，有时设计者希望链接的页面在新的窗口中打开，这时只要在<a>标签中添加“target=_blank”即可。程序 6.13 表现的是在新窗口中弹出页面的方法。

【本节示例参考：资料光盘\第 6 章\6-13　在新窗口中跳出链接窗口.html】

【实例 6-13】在新窗口中跳出链接窗口的方法，其源码展示如下：

程序 6.13　在新窗口中跳出链接窗口.html

```
01      <html>
02          <head>
03              <title>新窗口显示链接窗口</title>
04          </head>
05            <body>
06            <a href="后退.html"   target=_blank>新窗口显示链接窗口</a>
07            </body>
08      </html>
```

【运行程序】这个页面在浏览器中的结果如图 6.15 显示。



图 6.15　在新窗口中弹出链接窗口

6.5　案例：制作普通链接的主页

在网页上经常遇到这样的链接功能，一个主页上罗列很多名词条目，每个条目链接着一个页面，用来为专门的链接条目服务。下面的例子中介绍了美国电影史上票房最高的前 30 名电影。目前为止，泰坦尼克号依然排位第一，这里就以它为一个目录，运用各种链接方式。如程序 6.14、程序 6.15 展示了这个例子，类似的信息导航的页面有很多，通过这个引子来达到抛砖引玉的作用。

【本节示例参考：资料光盘\第 6 章\6-14　主页：电影 30 目录.html　6-15　子页面：泰坦尼克号.html】

【实例 6-14】主页：一个简单的电影目录展示，其源码展示如下：

程序 6.14　主页

```
01	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02	"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03	<html xmlns="http://www.w3.org/1999/xhtml">
04	<head>
05	    <style type=text/css>
06	        a {color:teal;                      //设置未访问时的链接颜色
07	          text-decoration:none              //去除链接下划线
08	          }
09	        a:visited {color:silver;            //设置访问后的链接颜色
10	                text-decoration:none
11	                }
12	        a:hover {color:red;                 //设置鼠标滑过链接时的颜色
13	                text-decoration:underline
14	                }
15	            </style>
16	        <!—使用样式表来修改页面链接状态-->
17	    <title>制作普通链接的主页</title>
18	    </head>
19	    <body>
20	    <h3>Imdb 电影评分排名</h3>
```

```
21          <p>
22          <a href=#1-10>1-10</a>                     //设置页面的链接
23          <a href=#11-20>11-20</a> 
24          <a href=#21-30>21-30</a>
25          </p>
26          <br><hr>
27          <p>
28          <a id="1-10"><a href="6-15 泰坦尼克号.html">  //设置页面的锚点
29                        1.    铁达尼号(1997)</a></a><br />
30          ......
31          </p>
32          <p>
33          <a id="11-20">11.    蜘蛛侠 2 (2004)</a> <br />
34      ......
35          </p>
36          <p>
37          <a id="21-30">21.    Iron Man (2008)</a> <br />
38           ......
39            </body>
40      </html>
```

【实例 6-15】子页面：泰坦尼克号，其源码展示如下：

程序 6.15　子页面

```
01      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03      <html xmlns="http://www.w3.org/1999/xhtml">
04        <head>
05          <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
06          <title>子页
07          </title>
08         </head>
09        <body>
10          <p>英文名：     Titanic</p>
11          <p><img src="图片/铁达尼克.jpg" width="97" height="140" /></p>
12        <p>中文名：  泰坦尼克号<br />
13        ......
14              <p>【剧情简介】 <br />
15             为了寻找 1912 年在大西洋沉没的泰坦尼克号和船上的珍贵财宝----价值连城的“海
16                   ......
17              <p><a href="6-14 主页.html"><img src="图片/返回主页.bmp"
18  border=0/></a></p>
19             </body>
20  </html>
```

【运行程序】浏览该页面，结果如图 6.16 所示。

既然是页面链接，首先必须是在两个以上的页面之间互动。这个例子中使用主页来罗列目录，如程序 6.14 所示，并且制作其中一个目录的子页，如程序 6.15 所示。如果单击“主页”中“1～10”、“11～20”或者“21～30”选项，页面会根据锚点位置，自动找到同页面中相对应的内容，如图 6.17 所示。单击“主页”中排位第一的“1. 铁达尼号（1997）”时，页面跳转到子页“泰坦尼克号”，在子页面下方的左边有个“小房子”图像，其作用是用来链接回主页。



图 6.16　页面跳转到锚点位置



图 6.17　子页链接回主页

在这个例子中，需要注意链接状态的设置，如程序 6.14 中，第 6～14 行，以及程序 6.14 中同页面内文本链接的设置和对“1. 铁达尼号（1997）”设置的超链接，而在程序 6.15 中，注意第 18 行代码，如何去除图像边框，并使用图像链接回主页的技巧。



注意：第 6～14 行使用的是 CSS 样式表来修饰链接状态，在第 7 章中有详细的介绍。在图 6.17 中由于主页的链接已经是被访问过的状态，所以呈现出设定好的灰色。

6.6　小　　结

本章介绍了超链接这一互联网中最具特色的技巧，这些技巧始终围绕在一个标签——<a>标签展开。但其展现出的形式却是多种多样，说明了超链接是人性化设计的一门技术，其变化非常多。本章主要的知识点有：

- 超链接的概念，以及如何确定链接地址放置页面文件。
- 基本的文本链接和图像链接。
- 链接到邮箱。
- 使用 id 属性的超链接，可以实现同一页面内的跳转。

- 使用 CSS 修饰链接的状态。
- 使用自定义图形和图像修改链接的下划线。
- 借助软件 Dreamweaver 制作图像热点区域链接。
- 在新窗口中跳出链接页面。

本章最后通过一个综合的例子表现了超链接的设置是一门自由的艺术。如果设计者愿意，可以将链接放在页面中的任何地方，但是不要忘记，提供友好、便捷的链接才是最重要的。在第 7 章中将介绍 CSS 的使用规则，通过 CSS 样式表，可以帮助设计者实现更多优秀的页面效果。

第 2 篇　页面制作提高篇

第 7 章　CSS 规则

第 8 章　表格

第 9 章　创建框架结构的页面

第 10 章　当 CSS 样式表遇到层

第 11 章　进一步讨论页面布局的方法



第 7 章　CSS 规则

对于设计者来说，总是希望能够在页面中自由发挥创意，实现自己的想法。然而，HTML 语言中的很多标签都存在很大的局限性，如<h1>标签定义的标题，始终是定义为较大的字体的文本，不会改变，所以在没有 CSS 以前，设计者不得不借助其他标签来补充标签的属性。而 CSS 是什么？CSS 就是一个无所不能的巨大的“属性集”。

从本章开始，读者要尽量忘记原来标签的定式。例如，虽然<h1>标签依然常被用来定义标题，但这是因为人们习惯于这样使用，而<h1>标签表现出来的样式却并非一成不变的，当设计者使用 CSS 时，他们可以令<h1>标签内的文本变成任何他们想要的样子。本章的知识点如下。

- 学习 CSS 样式表的写法规则。
- 理解选择器以及它的写法。
- 选择器的多种样式及它们的作用。
- 使用 CSS 样式表的 3 种方式，如行内 CSS、嵌入式 CSS 和外联式 CSS。
- 使用 CSS 样式表编辑页面。

7.1　如何学习 CSS

CSS 常被翻译为“级联样式表”，如果现在还不理解“级联”是什么概念，那么可以从 Cascading 一词的本意来理解。其有“小瀑布，瀑布状的”意思，不妨把 CSS 形象地理解为“瀑布一样的样式表”，一个 CSS 看上去是这样的：

```
body {
      font-family: 黑体;              //字体样式
      font-size: 80%;                 //字体大小
      color: black;                   //字体颜色
      background-color:blue;          //背景颜色
      margin: 1em;                    //在页面中的定位
      padding: 0;                     //设置空距为 0
      }
```

大括号中是一排属性值的描述，这样看上去是不是有些像瀑布状？其实，CSS 并不神秘，页面中，使用属性标签来修饰页面内容的表现。而 CSS 的作用正是用来修饰页面内容的表现形式的，所以 CSS 本身就是一个“属性集”，只不过这个属性可以由设计者自定义，可以无限扩展。

事实上，“级联”指的是 CSS 样式的继承性，在本章后面的内容中会体现出这一点。CSS 样式表有它自身的一定使用规则，这里通过一个简单的例子感受一下 CSS，如程序 7.1 所示为一个简单导航栏。

【本节示例参考：资料光盘\第 7 章\7-1　CSS 的使用.html】

【实例 7-1】CSS 的使用，其源码展示如下：

程序 7.1　CSS的使用.html

```
01        <html>
02          <head>
03            <title>使用 CSS 修饰导航栏</title>
04           <style>
…
…
21          </style>
22          </head>
23               <body>
24                 <div id="header">
25                 <h1>导航栏</h1>
26               <ul>
27                  <li><a href="#">目录 1</a></li>
28                  <li ><a href="#">目录 2</a></li>
29                  <li><a href="#">目录 3</a></li>
30                  <li><a href="#">目录 4</a></li>
31               </ul>
32                 </div>
33                 <div id="content">
34                  <p>使用 CSS 修饰导航栏</p>
35                 </div>
36               </body>
37             </html>
```

列表项

【运行程序】代码中缺少了第 5～20 行，这部分即 CSS 样式表定义部分。本例中先把这部分拿去，然后一步步地将 CSS 样式表填入，以观察 CSS 带来的变化。其中，<div>的作用是用来封装 CSS 样式表的，本书将在后面的章节中学习，缺少 CSS 的这段代码其结果如图 7.1 所示。



说明：本例用来展示 CSS 的效果，读者可以不必在意<div>标签所起的作用。

正如本书在前面章节中介绍的知识一样，这是一个普通的具备链接的列表。接着在原先的代码中补全第 5～9 行代码。而这几行代码就是一个简单的样式表，这个样式表中集合了 3 个属性，当补全到原来的代码中后页面变为如图 7.2 所示，导航栏的列表符号被去除了。

```
05        #header   ul {
06              list-style: none;      //去除导航栏的列表符号
07              padding:0;             //定位页面的位置
08              margin:0;
09                    }
```



图 7.1　未使用 CSS 的页面



图 7.2　去除列表符号

原来，这段代码是用来去除列表符号的，由“list-style: none;”这一属性中便可以看出来了。接着补全代码中的第 10～15 行。保存后重新打开这个页面，浏览器中新的结果如图 7.3 所示。

```
10          #header   li {
11                display: inline;                          //以行内形式展示列表
12                border: solid;                            //定义边框的样式
13                border-width: 1px 1px 0 1px;              //定义边框的粗细
14                margin: 0 0.5em 0 0;                      //定义其在页面中的位置
15                }
```

这几句代码又是用来改变列表的排列方式，比起原先的页面，已经发生了很大的改变。这里的 CSS 样式表和先前的相比，格式类同，改变的只是大括号中的属性。



注意：这个样式表的命名是 header li。这和<body>标签中的<li>又有什么关联呢？本章会在后面的内容中有所解释。

图 7.3 中的导航栏边框显得有些紧凑，而通过 CSS 能够很容易地改变边框的大小，这就需要用到 padding 属性。其是一个使用频度非常高的属性，作用是在一个声明中设置元素的内边距属性。在这里，给原来的代码补上第 16～18 行。最终在浏览器中更新的效果如图 7.4 所示。

```
16        #header li a {
17              padding: 0 1em;        //定义导航栏文本的链接
18                          }
```



图 7.3　横向排列导航目录



图 7.4　改变目录栏边框大小

当导航栏完成以后，接下来给页面中的文本“使用 CSS 修饰导航栏”添加边框，这在 HTML 标签中只能通过表格来实现，而且很麻烦。使用 CSS 样式表，只需要再加以下两句代码：

```
19    #content { border: 1px solid;            //定义边框的样式
20                    }
```

最后，整个页面就填补完成了，如图 7.5 所示是最终页面的效果。



图 7.5　给文本添加边框

【深入学习】不难发现，使用 CSS 时，原则是“先定义，再使用”。CSS 样式表的作用是用来表现页面，其可以令页面发生翻天覆地的变化，而 HTML 只需要完成构建基础的页面结构。

在这个例子中，CSS 的样式表都是放在<head>标签内的，注意样式表的定义规则。例如，#header ul 是如何和 HTML 标签相关联的，这其中有两个核心的问题，即如何定义 CSS 样式表和如何使用 CSS 样式表。那么围绕这两个问题，本章将详细介绍 CSS 的使用规则。

7.2　CSS 基本的规则写法

CSS 已经发展很多年了，但是直到前两年，CSS 才得以实现所有网页浏览器的支持。至今，CSS 已经形成了自己的一套语法，这套语法由一些标志构成，简单地说，就是一个基本的样式表由选择器、属性和属性值构成。

7.2.1　基本的样式表的写法

CSS 样式表也有属于自己的写法规则，在特定的规则下编写，才可以使之生效，一个标准的 CSS 写法是这样的，例如：

```
h1   {
    font-family:黑体;
     }
```

- h1：表示选择符。
- font-family：是属性，这里表示的是定义字体。
- 黑体：是属性值。这里表示定义的字体为黑体字。

“font-family:黑体;”将属性和属性值结合在一起，这样的形式称之为声明语句。声明语句可以有

很多句，所有的声明语句都要放在“{}”内。



注意：声明语句的结尾不能遗漏分号“;”。

h1 { font-family:黑体; }归结起来，就是一个基本的 CSS 样式表。

CSS 样式表的引用需要放在<style>标签中声明。

7.2.2 使用类 class 和标志 id 链接样式表

一个定义好的样式表，需要通过类 class 和标志 id 来定位它所作用的页面内容。id 标志在同页面中可以实现链接（参见第 6 章），作用相当于在页面中定位一个锚点。id 在链接 CSS 属性表时，所起到的作用也是一样的，而类似这样作用的还有类 class。类 class 和标志 id 之间是有区别的，如程序 7.2 所示为类选择器和标志选择器的对比。

【本节示例参考：资料光盘\第 7 章\7-2　类选择器和标志选择器.html】

【实例 7-2】类选择器和标志选择器，其源码展示如下：

程序 7.2　类选择器和标志选择器.html

```
01   <html>
02   <head>
03       <title>类选择器和标志选择器</title>
04     <style type="text/css">
05     .style1 {color: red;              //这是 class 选择器的定义样式
06             font-size:16px;
07             }
08     #style2 {color: blue;             //这是 id 选择器的定义样式
09             font-size:16px;
10             }
11    </style>
12    </head>
13     <body>
14       <h3 class="style1">使用 class 选择器 的红色字体</h1>
15       <h3 class="style2">使用 class 选择器 的红色字体</h1>
16       <h2 id="style1">使用 id 选择器 的蓝色字体</h2>
17       <h2 id="style2">使用 id 选择器 的蓝色字体</h2>
18     </body>
19   </html>
```

【运行程序】该例子中展示了类 class 和标志 id 是如何调用 CSS 样式表作用于 HTML 页面的，结果如图 7.6 所示。此外，类 class 和标志 id 之间在调用功能上也不同，class 可被用于多个对象，需要被设定成同种 CSS 样式的情况，假如这样写：

```
<h2 class="style1">
<h3 class="style1">
```

这种写法是正确的。但如果是 id 选择器，则只能用于一个对象。所以不能这样写：

```
<h2 id="style2">
<h3 id="style2">
```

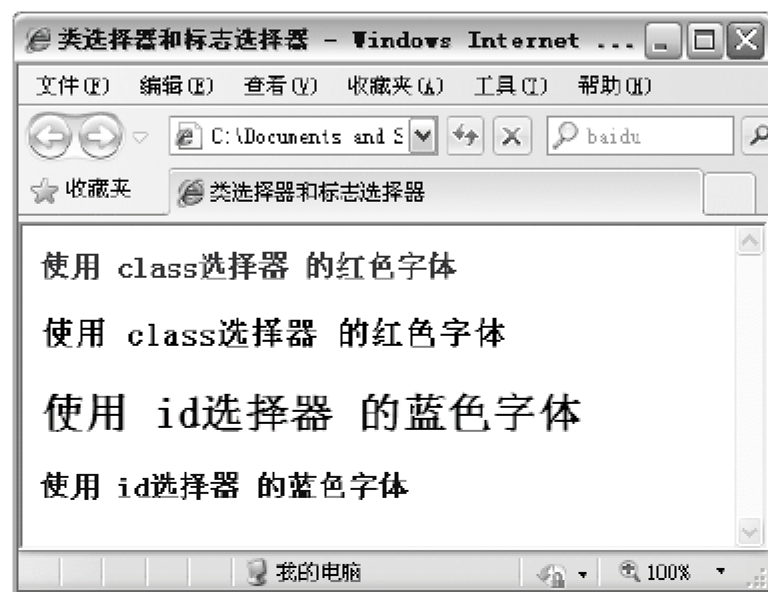


图 7.6　选择器

【深入学习】从图中可以看出，代码中第 14 行和第 17 行成功调用了 CSS 样式表，而第 15 行代码和第 16 行代码的运用是不起作用的。所以，从这个对比中可以看到，class 选择器和 id 选择器使用的方式是不同的。

7.2.3　创建选择器

选择器指定了样式将被应用于页面中的哪些内容，主要有 3 种基本选择器：HTML 选择器（HTML selector）、id 选择器（id selector）和 class 选择器（class selector）。通过这些基本的选择器，HTML 选择器可以延伸出派生选择器，而多种基本选择器混合使用时则定义为分组选择器。此外，还有一种特殊的伪类选择器。在本节中，将介绍这 6 种选择器。

1. HTML 选择器

所谓 HTML 选择器，其实就是重新定义 HTML 表现性标签的样式，正如文中最初所提的，<h1>标签内的文本应是黑色大体字，而当通过重新定义之后，<h1>标签内可以是任何样式，如程序 7.3 所示为如何将<h1>标签改变成一个选择器。

【本节示例参考：资料光盘\第 7 章\7-3　HTML 选择器的使用.html】

【实例 7-3】HTML 选择器的使用，其源码展示如下：

程序 7.3　HTML选择器的使用.html

```
01          <html>
02            <head>
03              <title> HTML 选择器的使用</title>
04             <style type="text/css">
05             h1 {
06                 color:#555555;                    //将文本颜色改变为灰色
07                 font-size:2.3em;                  //改变字体大小
08                 font-family: 微软雅黑;             //改变字体样式
09                 }
```

```
10          </style>
11        </head>
12        <body>
13          <h1> HTML 选择器的使用</h1>
14        </body>
15      </html>
```

【运行程序】浏览该页面，结果如图 7.7 所示。



图 7.7　HTML 选择器的使用

【深入学习】结果表明，<h1>标签下的文本已经不再是传统<h1>标签显示的文本，CSS 具有很便捷的灵活性，可以修改很多的标签。

2．派生选择器

通过依据元素在其位置的上下文关系来定义样式，称之为派生选择器。设计者可以使标记更加简洁，这种方式很好地体现了 CSS 的“级联”特性。下面这个例子使用了两个标记来定义 CSS 样式表。

```
h1 h2 {
  color:red;
  font-size:1em;
  font-family: 黑体;
    }
```



注意：“h1 h2”和“h1,h2”的含义是不同的，将在后面的分组选择器中介绍后者的定义。

在写法上，在上级对象 h1 和目标对象 h2 之间使用空格来指定样式，如将这个样式表添加入程序 7.3 的<style>标签中，样式表将作用于<h1>标签内的<h2>标签中的对象，这里添加以下几行代码来说明，如在程序 7.3 的<body>部分中放入以下代码：

```
<h1> HTML 选择器的使用</h1>
<h2> HTML 选择器的使用</h2>
<h1><h2> HTML 选择器的使用</h2></h1>
```

其结果如图 7.8 所示。

所以在 HTML 样式表定义时，当定义“h1 h2”样式表时，所起作用时并不是说 h1 和 h2 是两个相同的 CSS 样式表，而是被看作一个整体来定义。所以在页面中，使用“<h1><h2>”时，是作为一个整体出现的。因此，即使代码写成这样：

```
<h1>其他文本内容<h2> HTML 选择器的使用</h2>其他文本内容</h1>
```

其结果如图 7.9 所示。



图 7.8　标签符的级联一

其他文本内容

HTML选择器的使用

其他文本内容

图 7.9　标签符的级联二

只要是在“<h1><h2>”形式内的文本，无论<h1>和<h2>之间是否含有其他页面内容，一样都属于“<h1><h2>”样式表定义的范围。但是在<h1>标签中的文本，依然只受限制于 h1 样式表。

3．id 选择器

id 选择器的作用就是通过 id 选择器将 CSS 样式表作用到页面的对象上。那么这个样式表应该这样写，在选择器的开头处加上“#”符号。下面有这样一个样式表：

```
#text   { font-size:1em ;}
```

将这个样式表绑定到 HTML 对象上时，就要这样写：

```
<h1 id=#"text">文本内容</h1>
```

而 id 选择器一样可以作为派生选择器，代码如下所示：

```
#text p {color：blue }
```

这样，表明样式表将只作用于 text 对象中所有 p 标签下的内容。这和 HTML 样式表的嵌套原则是一样的。

4．class 选择器

如果希望通过 class 选择器将样式表固定到 HTML 页面对象，那么样式表的写法应该是在选择器的开头处加上“.”符号，例如：

```
.play { font-size:1em ;}
```

将这个样式表绑定到 HTML 对象上时，要使用 class 选择器，例如：

```
<h1 class="play">文本内容</h1>
```

和 id 选择器一样，class 也可以作为派生选择器，例如：

```
.fancy   td {
      color: #f60;                //修改文本颜色
      background: #666;      //修改页面背景颜色
            }
```



说明：td 标签表示的是表格单元。

在这个例子中，类名为 fancy 的元素内部的表格单元都会以灰色背景显示，文本则是橙色文字。

5．分组选择器

如果出现多个选择器定义的内容都是一样时，例如：

```
h1 {color:blue;}
#text {color:blue;}
.play {color:blue;}
```

那么，只要使用“,”隔开选择符就可以了。例如：

```
h1, #text, .play {color:blue;}
```



注意：当给选择符命名时，不能使用数字开头，必须以字母或下划线开头。

6．伪类和伪类选择器

伪类选择器并不是很多，它们通常用来定义一些特殊的效果。它们的写法由一个冒号和一个带有附加条件的属性组成，如超链接就是一个典型的伪类选择器（参照第 6 章）。此外，还有伪类“:lang（语言）”，它的作用是允许设计者为不同的语言定义特殊的规则。如程序 7.4 所示为使用的伪类的其中一种。

【本节示例参考：资料光盘\第 7 章\7-4　伪类“:lang（语言）”的使用.html】

【实例 7-4】伪类“:lang（语言）”的使用，其源码展示如下：

程序 7.4　伪类“:lang（语言）”的使用.html

```
01          <html>
02            <head>
03              <style type="text/css">
04              q:lang(smile)
05              {
06              quotes: "o(∩_∩)o" "~"      //这里定义了将 smile 转换的符号
07              }
08              </style>
09            </head>
10            <body>
11              好吧，展示一个笑脸吧
12              <p>博君一笑 <q lang="smile"> 祝你愉快 </q></p>
13            </body>
14          </html>
```

【运行程序】浏览该页面，结果如图 7.10 所示，伪类代码的位置被伪类定义的内容所替代。



图 7.10　伪类“:lang（语言）”的使用

> **注意：使用伪类定义的页面可以使用 Firefox 浏览器打开，而 IE 7.0 之前的版本均不能有效支持。**

【深入学习】CSS 中还有一些其他伪类，如表 7.1 所示。

表 7.1　CSS中的伪类

伪　　类	作　　用
:active	将样式添加到被激活的元素中
:focus	将样式添加到被选中的元素中
:hover	当鼠标悬浮在页面对象上方时，向页面对象添加样式
:link	将样式添加到未被访问过的链接中
:visited	将样式添加到被访问过的链接中
:first-child	将特殊的样式添加到页面对象的第一个子元素中
:lang	允许设计者定义指定的页面中所使用的语言

7.2.4　应用 CSS 样式表

首先需分清这样一个概念，应用 CSS 样式表到 HTML 页面中，和将 CSS 样式表绑定到 HTML 页面中的对象，这两者是不同的两个概念。例如，在 7.2.3 小节中，通过不同的选择器将样式表绑定到 HTML 页面中的对象。但是，使用的都是同一种方法——应用 CSS 样式表到 HTML 页面中，这种方式称之为嵌入样式表，而这种应用 CSS 的方法亦有 4 种，剩下的 3 种方法分别是内联样式表、外联样式表和多重样式表。

1．嵌入样式表

嵌入样式表是指使用<style>标签将 CSS 样式表放入到<head>标签内（参照 7.2.3 小节中的示例）。这种用法的好处在于将页面的表现性和结构性实现很好的分离。对于设计者来说，使共同协作的效率更高，写法是：

```
<style type="text/css">
…
</style>
```

<style>标签内部是 type 属性。当然，也有一些其他的属性可供使用，如 media 属性，可以用定义以何种媒介来提交文档。文档可以被显示在显示器、纸媒介或者听觉浏览器中，如下代码所示：

```
<style type="text/css" media="screen">
```

2．内联样式表

同时又常被称为行内 CSS，通过使用 style 属性来引用声明语句，放在页面结构性标签的后面，例如：

```
<p style="color: sienna; margin-left: 20px">
页面文本内容
</p>
```

但是在页面设计时，内联样式表应尽量少用，因为这种方法不能将页面表现和页面结构很好地分开，通常是在不得已的情况下，用作补充调整的作用。

3．外联样式表

在创建整站的工程时，整站包含很多不同的子页面，设计者希望一个 CSS 样式表可同时用于几个页面。这时使用外联样式表是非常好的选择。因为 CSS 样式表可以作为一个独立文件存储在页面外部，后缀名为“.css”。通常的情况是使用<link>标签来使用样式表，写法是：

```
<link rel="Shortcut " type="text/css " href="some。css " />
```

当然，超链接的不一定是一个文件，也可以引用 URL 地址文件。此外还可以使用“@import”来应用一种外联样式表，把它放在 style 标签中间，写法如下：

```
<style type="text/css">
@import url("some.css");
</style>
```



注意：@import 是 CSS2 的时候提出来的，所以早期的浏览器并不支持，如 IE 5.0 之前的浏览器。

4．多重样式表

在样式表中，如果出现多种样式表，它们之间总有个先后的问题。通常来说，当多个样式表作用于同一个页面对象时，离这个页面对象最近的样式表起决定作用。但是，行内样式表始终处于最高级别，如程序 7.5 所示为多重样式表下的先后关系。

【本节示例参考：资料光盘\第 7 章\7-5　多重样式表情况下的优先级.html】

【实例 7-5】多重样式表情况下的优先级，其源码展示如下：

程序 7.5　多重样式表情况下的优先级.html

```
01          <html>
02            <head>
```

```
03              <title>多重样式表情况下的优先级
04              </title>
05          <link rel= "stylesheet" type="text/css" href="bobo.css">        //这是外联样式表
06          <style type="text/css">
07            .hui {color:red;
08                font-size:20px;                                          //这是内联样式表
09                }
10            </style>
11          </head>
12          <body>
13          <div class="bobo">
14          <p class="hui">多重样式表情况下的优先级
15          </div>
16              <div class="hui">
17              <p class="bobo">多重样式表情况下的优先级
18              </div>
19                  <div class="hui">
20                  <p style="color:orange"; class="hui";>多重样式表情况下的优先级
21                  </div>
22          </body>
23        </html>
```



说明：在这个例子中，使用层 div 封装了 3 个部分，在层之内的样式表不会影响到层之外的文本。

【运行程序】.bobo 是外部样式表，这里把.bobo 外部样式表定义为：

```
.bobo {color:blue;
      font-size:20px;
   }
```

那么，结果如图 7.11 所示。

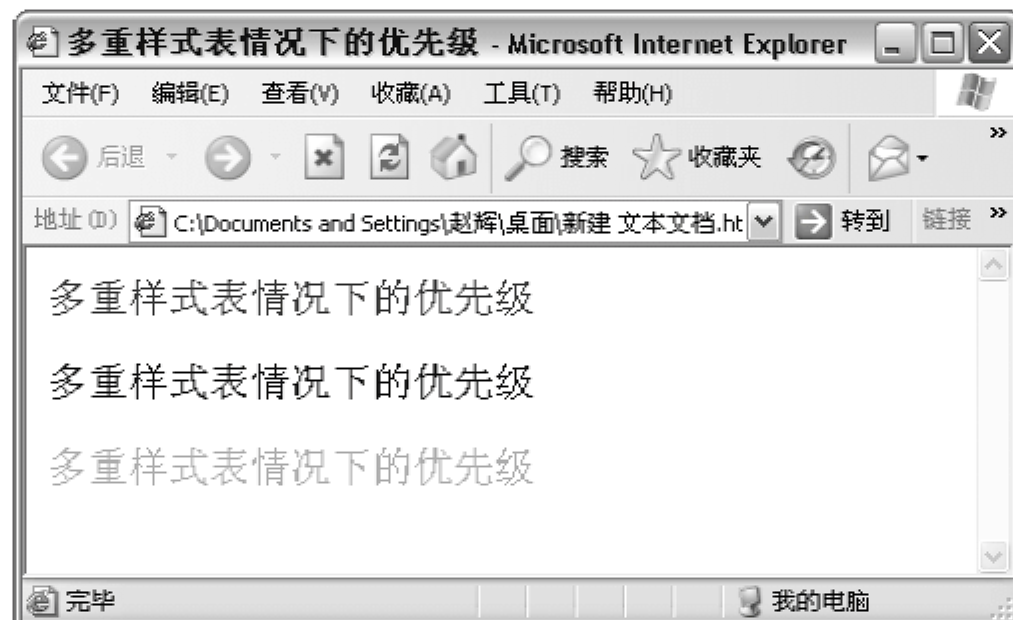


图 7.11　多重样式表情况下的优先级

【深入学习】可见，在第 14 行和第 17 行代码中，两行的样式表交换了位置，而作用于文本的则

是离页面对象最近的样式表。而在第 20 行代码中，起决定效果的是行内链式表。所以，同一种引用样式表的情况下，离页面对象最近的样式表起作用。

7.3 用 CSS 来修饰页面文本

本书在前面的章节中介绍了文本的排版，并没有花浓重的墨彩去描绘如何展现文本的形式，而当理解了 CSS 的使用后，这里将为设计者打开一扇广阔的设计天空，令文本可以变化出奇妙的样式。

7.3.1 修饰页面文本字体

不要忽视字体的作用，字体的设置能潜移默化地给浏览用户带来很大的视觉影响。在 CSS 中，使用 font-family 属性来定义字体的样式。说到字体，中国文化中的字体一向渊源博大，至少也可以定义出上百种。但是可惜，原则上浏览器只支持系统中默认的字体。如果设计者希望使用自定义字体，需要先将字体存入电脑中。如程序 7.6 所示为通过样式表来修饰文本字体。

【本节示例参考：资料光盘\第 7 章\7-6 CSS 修饰文本字体.html】

【实例 7-6】CSS 修饰文本字体的方法，其源码展示如下:

程序 7.6 CSS修饰文本字体.html

```
01    <html>
02      <head>
03        <title>CSS 修饰文本字体</title>
04       <style type="text/css">
05       h1 {font-family:微软雅黑;              //设置标题字体样式
06           }
07       p {font-family: 隶书;                  //设置段落字体样式
08           }
09       </style>
10      </head>
11      <body>
12       <h1>2008 中国经济发展形式</h1>
13        <p>
14         从宏观上说，中国经济发展整体形势良好。2008 年国家统计局公布数据为上半年
15    国内生产总值 130619 亿元
16    ......
17        </p>
18      </body>
19    </html>
```

【运行程序】浏览该页面，结果如图 7.12 所示。

【深入学习】这样就完成一个简单的页面文本的修饰了，只是稍稍地做了一点字体改变，但整个页面看起来已经略有不同，似乎是一张散发油墨的报纸。



图 7.12　使用 CSS 修饰文本字体



注意：如果不确定浏览器中的字体，或者想尝试一些罕见的字体，可以在指定的字体后添加备用字体，例如：

```
font-family: "全新硬笔行书简",宋体;
```

这样，如果浏览者系统中没有“全新硬笔行书简”这个字体，浏览器会默认改变页面文本为宋体。

7.3.2　文本的字号

font-size 属性用来改变字体的大小。在规定字体大小时，最常见的有 3 种表示方法，也可以说是 3 种度量方案，分别是 px、em 和%。

- 像素单位 px：使用像素直接来定义字体的大小，绝对单位，如 12px，无论在哪个显示器中，字体的大小只会相对于显示设备来确定。
- 字体大小 em：这是目前较为流行的一种定义字体大小的方式，一个字体的大小就是 1em，任何浏览器的默认字体大小都是 1em。用户可以根据自己的喜好定义默认字体，网页中的字体大小就可以弹性地随着改变。
- 百分比 %：这是以当前文本的百分比定义尺寸。例如，{ font-size:200%}是指文字大小为原来的两倍。在页面制作中，通常使用%定义页面主体的初始字体大小，然后以 em 为单位表示字体的大小，如程序 7.7 所示为改变页面文字的大小。



注意：还有一种表示单位的方式：pt，12pt=1em。

【本节示例参考：资料光盘\第 7 章\7-7　CSS 修饰文本字体大小.html】

【实例 7-7】CSS 修饰文本字体大小，其源码展示如下：

程序 7.7　CSS修饰文本字体大小.html

```
01        <html>
02          <head>
03             <title>CSS 修饰文本字体大小</title>
```

```
04              <style type="text/css">
05               body {font-size:80%;
06                     }                                //将整个页面的字体定义为标准的 80%
07               h1 {font-family:微软雅黑;
08                   font-size:2em;                     //将页面文本字体的大小定义为 2em
09                    }
10               p {font-family: 隶书;
11                  font-size:1.5em;
12                  }
13              </style>
14             </head>
15              <body>
16               <h1>2008 中国经济发展形式</h1>
17                <p>
18     从宏观上说，中国经济发展整体形势良好。2008 年国家统计局公布数据为上半年国内生产总
19     值 130619 亿元，
20          …
21              </body>
22          </html>
```

【运行程序】浏览该页面，结果如图 7.13 所示。

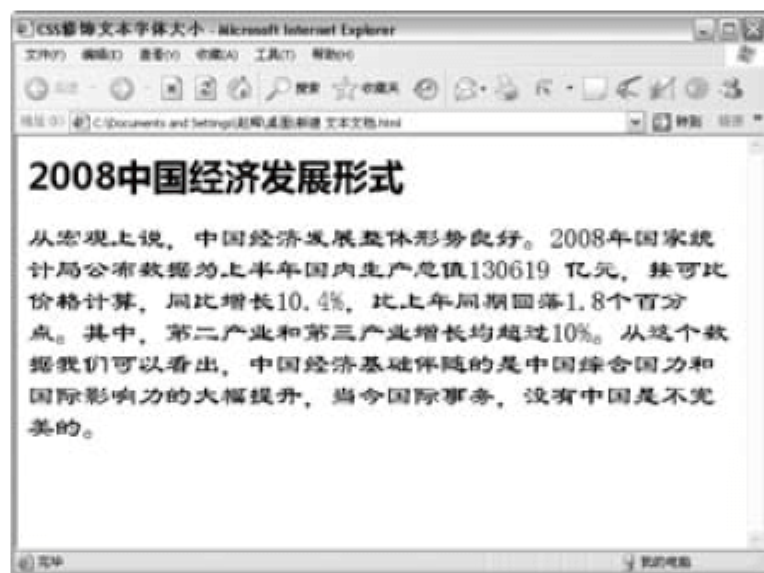


图 7.13　CSS 修饰文本字体大小

【深入学习】注意和图 7.12 相比，这个页面的文本大小更合理，字体配合页面也更加舒适，使页面效果显得十分协调。所以在设计页面文本时，要注意标题和文章内容的大小对比，还有在整体页面中的效果。这不仅是技术上的问题，需要在不同的情况下作出灵活的应变，而 CSS 很好地提供了舒适的自由度。

7.3.3　文本段落行高

使用 line-height 进行行高的设定，同样，段落行高的数值也可以通过单位 px、单位 em 和百分比来设定。1em 和 100%代表正常的行距，所以用 em 和%表示行距也是比较好的选择，如程序 7.8 所示。

【本节示例参考：资料光盘\第 7 章\7-8　不同行高的对比.html】

【实例 7-8】不同行高对比的方法，其源码展示如下：

程序 7.8　不同行高的对比.html

```
01          <html>
02              <head>
03                <title>CSS 修饰文本字体大小</title>
04              <style type="text/css">
05                body {font-size:80%;                          //设置页面字体的大小
06                      }
07                h1 {font-family:微软雅黑;
08                    font-size:2em;                            //设置标题字体的大小
09                      }
10                .big {font-family: 隶书;
11                      font-size:1.5em;
12                      line-height:1.5em;                      //设置行高为 1.5em
13                      }
14                 .small {font-family: 隶书;
15                        font-size:1.5em;
16                        line-height:0.8em;                    //设置行高为 0.8em
17                        }
18                 .normal {font-family: 隶书;
19                          font-size:1em;                      //设置行高为 1em
20                          }
21              </style>
22             </head>
23              <body>
24              <h1>2008 中国经济发展形式</h1>
25              <p class="big">从宏观上说，中国经济发展整体形势良好。2008 年国家统计局公布数据为上半年
26     国内生产总值 130619 亿元，
27          ……
28              <p class="small">按可比价格计算，同比增长 10.4%，比上年同期回落 1.8 个百分点。其中，
29     第二产业和第三产业增长均超过 10%。
30            <p class="normal">从这个数据我们可以看出，中国经济基础伴随的是中国综合国力和国际影响力
31     的大幅提升，当今国际事务，没有中国是不完美的。
32              </body>
33          </html>
```

【运行程序】浏览该页面，结果如图 7.14 所示。



图 7.14　CSS 修改段落行距



注意：图中 3 段文本行距依次对应为 1.5em、0.8em 和 1em。

【深入学习】留意代码中第 10～20 行，这部分是对页面中的段落文章进行设定，但是未免有些拖沓，更好的方式应该写成：

```
p {font-family: 隶书;
    font-size:1.5em;
   }
p.big { line-height:1.5em
      }
p.small{ line-height:0.8em
  }
```

使用分组选择器样式来精简代码

这样使用分组样式定义，才真正意义上做到了精简页面的代码，灵活地使用了 CSS 样式表。

7.3.4 禁止文本自动换行

大部分互联网上的页面都禁止页面内容自动换行，这似乎已经成了一种默认的许可。而如果页面随着浏览器的大小自动换行确实令人眼花缭乱，既然人们已经习惯于这样的页面展示形式，那么就将这个习惯保持下去。通过 white-space 属性可以禁止文本自动换行，如程序 7.9 所示为禁止文本自动换行的页面。

【本节示例参考：资料光盘\第 7 章\7-9　禁止文本自动换行.html】。

【实例 7-9】禁止文本自动换行的方法，其源码展示如下:

程序 7.9　禁止文本自动换行.html

```
01          <html>
02            <head>
03            <title>禁止文本自动换行</title>
04            <style type="text/css">
05              p { white-space: nowrap      //nowrap 属性指定页面无法自动换行
06                }
07            </style>
08            </head>
09              <body>
10          <p>历史书上写道：三国吴王孙权派 1 万官兵到达“夷洲”（台湾），吴人沈莹的《临海水土志》留下了
11    世界上对台湾最早的记述。隋唐时期（公元 589—618 年）称台湾为“流求”。隋王朝曾三次出师台湾。
12             </body>
13          </html>
```

【运行程序】浏览该页面，结果如图 7.15 所示。



图 7.15　禁止文本自动换行

所以在这个页面中，当使用 white-space 时，浏览器中的页面不会因为窗口大小而自动换行，因而出现下滑栏。



说明：此外，当定义为 white-space: pre 时，其作用相当于<pre>标签（参照第 3 章内容）。

7.4　给页面对象添加颜色

在第 5 章中已经介绍了颜色的基本概念。而 CSS 所要做的工作，就是通过不同的属性作用于不同的 HTML 页面对象，将颜色值赋予这些对象就可以了。主要有 color 和 background-color 这两个重要的属性。准确地说，color 属性修改其对象前景色，而 background-color 则修改其对象的背景色。通过下面这个例子可以很好地说明前景色和背景色的概念，如程序 7.10 所示为使用 CSS 修饰页面的颜色。

【本节示例参考：资料光盘\第 7 章\7-10　使用 CSS 修饰页面颜色.html】

【实例 7-10】使用 CSS 修饰页面颜色的方法，其源码展示如下：

程序 7.10　使用CSS修饰页面颜色.html

```
01    <html>
02      <head>
03        <title>使用 CSS 修饰页面颜色</title>
04      <style type="text/css">
05      body {color:white;
06            background-color:black;        //设置页面背景颜色为黑色
07            font-family:微软雅黑;           //设置字体的样式
08            font-size:1.2em;               //设置字体的大小
09            }
10      h2 {background-color:green           //设置文本背景颜色为绿色
11         }
12      p {background-color:orange;          //设置文本背景颜色为橙色
13        }
14      </style>
15      </head>
16      <body>
17          <h2>多重样式表情况下的优先级</h2>
```

```
18                <p>多重样式表情况下的优先级</p>
19            </body>
20        </html>
```

【运行程序】浏览该页面，最终的浏览结果如图 7.16 所示。



图 7.16　使用 CSS 修饰页面颜色

【深入学习】如代码第 5 行所示，body 样式表定义了页面背景是黑色，而文本颜色是白色。然而也要注意代码第 10 行和第 12 行，分别定义了各自的对象背景色是“绿色”和“橙色”。可以发现，文本的背景部分变成了对应于不同 CSS 样式表作用的颜色，分别是“绿色”和“橙色”。CSS 中，关于文本和页面背景的属性很多，这里将它们列举出来，如表 7.2 所示。

表 7.2　修饰页面文本和页面背景的属性

属　性	说　明
background	作用是将背景属性设置在一个声明中
background-color	设置页面对象的背景颜色
background-image	引用图像设置为背景
background-repeat	设置背景图像重复的方式
background-position	设置背景图像的具体位置
background-attachment	背景图像是否固定或者随着页面的其余部分滚动
color	设置文本颜色
line-height	设置行高
white-space	设置元素中段落排版的方式
word-spacing	设置字间距
font-family	设置文本字体
font-size	设置字体的尺寸
font-style	设置字体风格
font-weight	设置字体的粗细
direction	设置文本方向
letter-spacing	设置字符间距
text-align	对齐页面中的文本
text-decoration	给文本添加下划线
text-transform	控制元素中的字母

7.5　案例：使用 CSS 制作个人页面

使用样式表可以更容易地做出比较个性化的页面，良好的习惯是先在纸上画出大概页面的草图，根据草图，制定需要的 CSS 样式表。当然，还要准备好页面的文字资料和图像。当这些工作都完成之后，接下来就是将这些素材拼接在一起。在这个例子中，表现的是一个个人主页，并没有使用复杂的技术，仅仅是通过使用图像、放入文本这样的技术，如程序 7.11 所示为制定的个人页面。

【本节示例参考：资料光盘\第 7 章\7-11　个人页面.html】

【实例 7-11】个人页面，其源码展示如下：

程序 7.11　个人页面.html

```
01	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02	"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03	<html xmlns="http://www.w3.org/1999/xhtml">
04	  <head>
05	    <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
06	    <title>使用 CSS 制作个人页面</title>
07	  <style type="text/css">
08	   body {background-image:url(图片/背景.png);
09	         background-attachment:fixed;           //设置页面背景图像放置方式
10	         font-size:100%;
11	         }
12	    p {text-align:right;
13	       }
14	   .biaoti { text-align:right;
15	             color : #D35C0F;
16	             font-size : 2.5em;                  //设置字体的大小
17	             font-weight : bold;
18	             line-height : 1.4em;                //设置行高
19	             font-family : Cooper Std Black;
20	             }
21	    #date { text-align:right;                    //设置文本对齐方式
22	              color : #999999;                  //设置文本颜色
23	              font-size : .8em;
24	              font-weight : bold;
25	              font-family : Geneva, Arial, Helvetica, sans-serif;
26	              }
27	    #content {color : #3399FF;                   //设置文本颜色
28	                font-size : 1em;
29	                line-height : 1.6em;             //设置行高
30	                font-weight : bold;
31	                font-family :幼圆;
```

```
32                  white-space: pre;
33              }
34          a:link   {color : #6d2542;
35                   text-decoration : none;
36                   }
37          a:visited {color : #999999;
38                   text-decoration : none;
39                   }
40          a:hover {color : #999999;
41                   text-decoration : underline;
42                   }
43          a:active {color : #999999;
44                   text-decoration : none;
45                   }
46        </style>
47        </head>
48        <body >
49      <p><img src="图片/标题.png" alt="标题" height="120" align="right"/>
50      <p class="biaoti">JennyYuan
51      <p id="date"><a href="后退.html">2008 年 9 月 12 日</a></p>
52      <p><hr align="right" width="45%" >
53      <p id="content">
54          把掌声留给坚守得住承诺的人.
55          毕竟,为了承诺一直守侯.
56          多辛苦,多执着,多浪漫.
57  …
58          The End 在第 100 天。
59        </body>
60      </html>
```

通过伪类修改页面的链接状态

【运行程序】代码的第 1～47 行部分是页面的头部，主要包括声明代码和 CSS 样式表的制定，而这些制定都是建立在已确定好的页面结构上。第 48～60 行的部分是页面的结构代码。结果如图 7.17 所示。



图 7.17　使用 CSS 制作个人页面

【深入学习】在这个例子中，页面结构非常简单，由代码可以知道，几乎都是<p>来排版的，大部分都是 CSS 帮助设计者来实现的效果。这样的页面在设计时要注意字体、字号和行距的设定，不要忽视了这些小细节，它们带来的影响是非常重要的。

代码第 51 行是样式表和伪类同时作用于一个对象。整个页面包含了行内样式表如第 49 行和第 52 行，以及嵌入式样式表的引用如代码第 50 行、51 行和 53 行。不同样式表的引用互相配合，相得益彰。

其中，第 8～13 行属于 HTML 页面选择符，第 14～20 行是类选择符，第 21～33 行是标志选择符，第 34～45 行是典型的伪类选择符，用于设置页面中的链接状态。

当然，这个页面代码并非十分理想。由于 CSS 样式表的级联性，样式表之间容易由上级对下级产生影响，这样在编排页面时总得小心翼翼。因此，在页面设计时便引入了 div 层的概念。层的概念是用来封装 CSS 样式表，避免 CSS 样式表之间出现错误交叉。



说明：层的学习将在后面的章节中详细介绍。

7.6　小　　结

本章是关于 CSS 样式表的学习，涉及到的概念颇多，需要读者用心去理解，在理解的基础上才能对 CSS 运用自如。本章需掌握的知识点有：

- CSS 的写法规则，包含选择符、属性和属性值，如“color:red;”。
- 选择符的 3 种写法，一般的选择符、以“#”开头的 id 选择符和以“.”开头的 class 选择符。
- 选择器的 6 种样式及它们的作用，HTML 页面选择器、id 选择器、class 选择器、派生选择器、分组选择器和伪类选择器。
- 使用 CSS 样式表的 3 种方式，即行内 CSS、嵌入式 CSS 和外联式 CSS。
- CSS 中文本属性的介绍。

在本章的最后给出了一个综合例子，结合各种样式表制作了一个个人页面。在第 8 章中，将介绍如何在页面中设计表格，通过表格来表现页面的内容。

第 8 章　表　　格

没有 CSS 之前，表格是最流行的布局页面方式，而不只是单元格组成的表格，不过那已经是很久以前的事情了。现在谈论 Web 设计时，再说到“表格”，那么它就是一个普通的由单元格组成的表格。

生活中经常需要用到表格，同样，虽然现在表格的使用频率大不如前，但它又是页面设计中不可或缺的一个元素。在一些服务性网站中，经常需要使用表格来传递信息给浏览者，方便用户下载，如电信公司的话费单等。当然，如果你的老板要求你整理工作报表，你告诉他在你的个人主页上提供每个月的报表下载，说不定也是一件很酷的事情。本章的知识点如下。

- 学习制作表格。
- 修饰表格的单元格。
- 拆分合并表格。
- 使用 CSS 样式表来表现表格。
- 表格的实际应用。

8.1　理解页面中的表格

表格看上去虽然只是一个一个的小格子组成的样子，但是，谈及在 HTML 中制作表格，远远不是看上去那么直接。表格涉及的属性很多，因为人们在表述表格时，不是说“某某表格左上角的那个格子”，而是通过描述某一行和某一列来定位某个单元格的位置，这里已经描述了两个属性。一个普通的表格在页面中看上去如图 8.1 所示。

图 8.1　一个普通的表格

这是一个 3×3 的表格，横排为行竖排为列，每一个格子称之为单元格。在 HTML 页面中不仅可以定义表格行列的数量，还可以设定表格边框的样式、单元格的长宽高，以及单元格彼此之间的距离。

所以在 HTML 页面中，编辑表格是一项不困难却很繁琐的工程。

8.2　普通表格的应用

这种表格常见于课程表、出勤表或者价目表这种形式的表格，使用频率很高。因此，如果只是针对简单的文本内容，仅仅是希望这些内容横排或者竖排，那么表格是一个比较好的方法。

8.2.1　制作普通表格

表格属于结构性标签，使用<table>标签。一个最简单的完整表格也需要具备表头、行、列的信息。所以它的代码是这样的：

```
<table >
   <tr>
     <th>Head1</th>
     <th>Head2</th>
   </tr>
   <tr>
     <td>row 1, cell 1</td>
     <td>row 1, cell 2</td>
   </tr>
   <tr>
     <td>row 2, cell 1</td>
     <td>row 2, cell 2</td>
   </tr>
   …
</table>
```

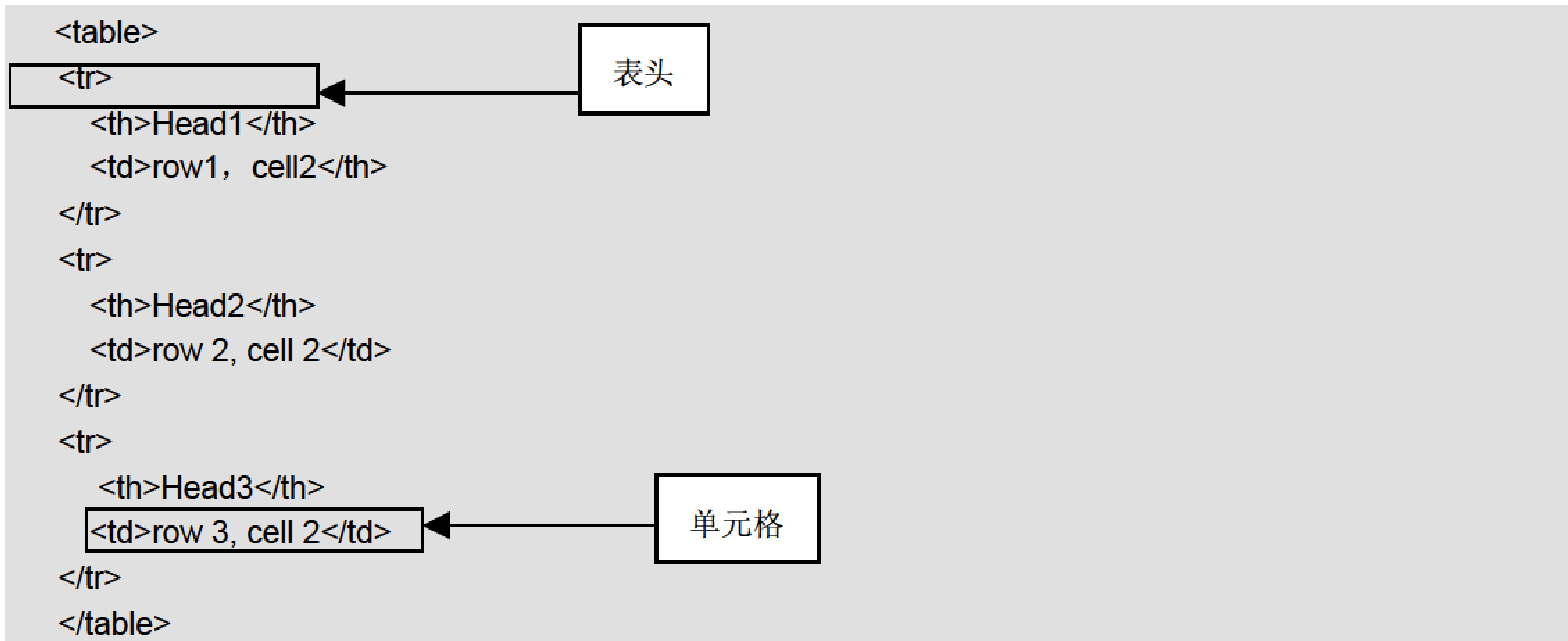
这样看代码真的很难想象这个表格是什么样子，如果用表格的形式去表达这段代码，则如图 8.2 所示。

通过和表格实体的对比，可以明白此前的代码表示的是一个 3×2 的表格，而<table>标签封装了一个表格。在 HTML 中，首先用<tr>来定义行，如这里有 3 行，<table>标签中即先定义 3 组<tr>标签。接着用<td>来定义每一个单元格，<th>表示的是表头单元格，所以，不仅每一行都要描述清楚，而且每一个单元格也要逐一定义。



说明：如果不希望有表头行，可以省去<th>标签。此外，IE 浏览器会默认隐藏空单元格，即如“<td></td>”。如果中间没有内容，最后显示的效果会很奇怪。习惯地最好放入一个空格占位，如“<td> </td>”。

当然，表格的代码也可能写成如下代码，这是以列为表头的表格，其结果如图 8.3（上边）所示。



或者也可以写成如下代码，将第一组的<tr>标签内的对象全部定义为表头。此后每一组<tr>标签内的第一个标签定义为<th>标签，即之后的每一行第一个单元格表示为表头。这样就是分别以横列第一行作为表头。

```
<table>
 <tr>
    <th>Head1</th>
    <th>HEAD1</th>
 </tr>
 <tr>
    <th>Head2</th>
    <td>row 1, cell 2</td>
 </tr>
 <tr>
    <th>Head3</th>
    <td>row23, cell 2</td>
 </tr>
</table>
```

这样每一个行的第一个单元格被认为是表头，如图 8.3（下边）所示。

tr	th Head1	th Head2
tr	td row 1, cell 1	td row 1, cell 2
tr	td row 2, cell 1	td row 2, cell 2

```
<tr>
  <th>Head1</th>
  <th>Head2</th>
</tr>
<tr>
  <td>row 1, cell 1</td>
  <td>row 1, cell 2</td>
</tr>
. . .
```

图 8.2　表格的代码形象化比较

以列为表头的样式，

Head1	row1，cell2
Head2	row 2, cell 2
Head3	row 3, cell 2

以横列均为表头的样式，

Head1	HEAD1
Head2	row 2, cell 2
Head3	row 3, cell 2

图 8.3　表头的样式

8.2.2　表格的基本属性

表格的基本属性即是指表格的行、列和单元格，但并不是每一个表格的单元格都是统一大小的，所以需要设计者通过一些属性参数去修改表格的样子，让它们可以变得更多样性一些。这样的属性有行高、宽度等。

1．行高 height 属性

由于默认情况下一个空白表格的单元格是平均分配的，所以如果需要自定义行高，要事先设定表格的总高度。可以使用 CSS 样式表先定义 table，然后定义 th 或 tr。若想改变图 8.2 中表格的表头行高，则样式表应该写为：



注意：表格中的内容会自动影响到单元格的大小分配。

```
<style type="text/css">
table {height:185px;
}
table th {height:32px;
}
</style>
```

当使用这个 CSS 时，结果如图 8.4 所示。

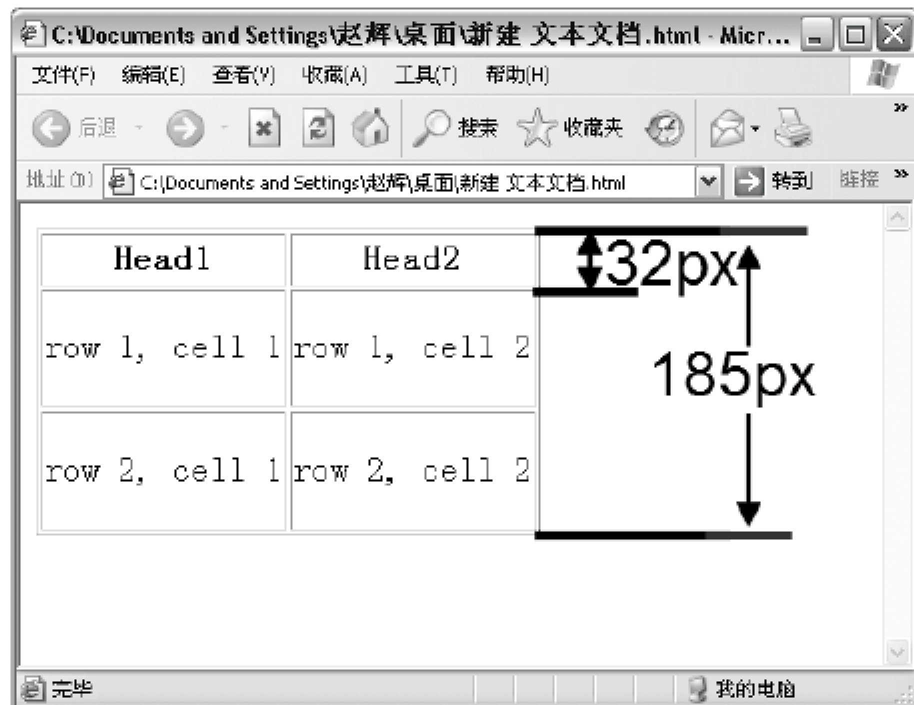


图 8.4　表格行高



注意：因为表格中只有一个表头，即<th>，所以才可以定义样式表为“table th{}”。如果要定义普通行<tr>，那么应该选择其他定义方式，如可以定义为“#tr1 ()”，通过“<tr id="r1">”绑定到对象。

2．宽度 width 属性

如果只是需要修改表格列的宽度，则只使用 width 属性即可。但是不同于行高的是，表格中的宽度是针对整个表格或每一个单元格的，所以像下面这样的写法是错误的。

```
table {width:400px;
      }
table th {width:100px;
        }
```

表格的宽度并不会改变。这是因为表格中一行有多个单元格，浏览器无法辨认解析代码指定的具体是哪一个单元格的宽度。而由于这一行无论有多少个单元格，它们的行高始终是相同的，所以设置行高时不会有问题。那么为了避免这样的错误，这里使用一种更方便的方法，如程序 8.1 所示为修改表格的宽度。

【本节示例参考：资料光盘\第 8 章\8-1　修改表格的宽度.html】

【实例 8-1】修改表格宽度的方法，其源码展示如下：

程序 8.1　修改表格的宽度.html

```
01    <head>
02      <title>设定表格的宽度</title>
03      </head>
04      <body>
05        <table height="168" border="1">
06          <tr >
07            <td width="144">Head1</th>      //通过表头来设置表格单元格的宽度
08            <td width="240">Head2</th>
09          </tr>
10          <tr >
11            <td >row 1, cell 1</td>
12            <td>row 1, cell 2</td>
13          </tr>
14          <tr >
15            <td >row 2, cell 1</td>
16            <td>row 2, cell 2</td>
17          </tr>
18        </table>
19    </body>
```

【运行程序】这样表格宽度就改变了，如图 8.5 所示。

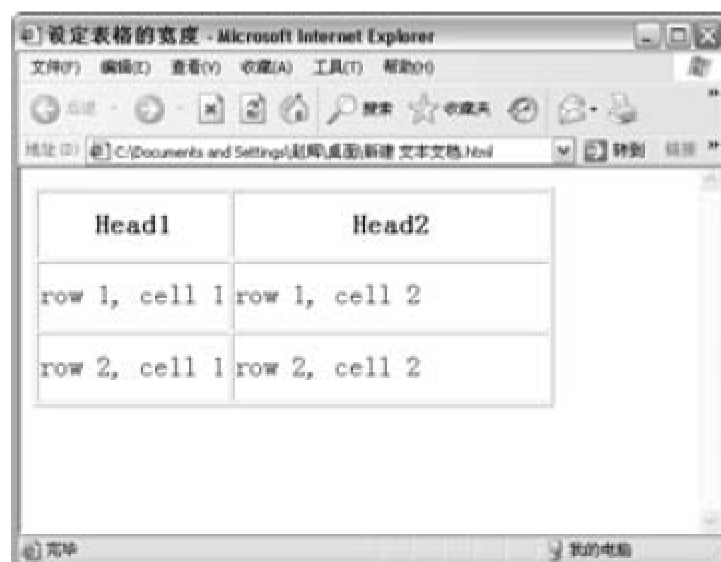


图 8.5　修改表格的宽度

3．单元格大小属性 height 和 width

单元格的大小其实就是由高和宽两个因素组成的。所以，如果要准确定位一个单元格的具体大小，这两个因素是缺一不可的，必须要同时具备，这样才能定位每个单元格的大小。



说明：修改行高时使用了 HTML 页面选择器，修改宽度时选择使用了页面内修改表格的属性。在这两种方法中，后者更适合使用。

8.2.3　合并单元格

合并同行单元格使用 colsplan 属性。在需要修改的那一行中，先删除多余的单元格，这是重要的一步，如果删错了单元格，很可能最后呈现的表格会面目全非。这之后再定义合并的单元格。合并同排的单元格使用 rowspan 属性，这里介绍一种合并单元格的方法，如程序 8.2 所示为合并单元格的范例。

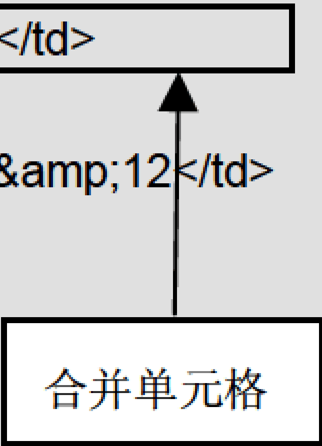
【本节示例参考：资料光盘\第 8 章\8-2　合并单元格.html】

【实例 8-2】合并单元格的方法，其源码展示如下：

程序 8.2　合并单元格.html

```
01  <html>
02    <head>
03      <title>合并表格中单元格</title>
04    </head>
05    <body>
06    <h3>普通的表格</h3>
07    <table width="305" height="156" border="1">    //设置表格的长宽和边框的粗细
08    <tr>
09      <td width="70" height="50">1</td>
10      <td width="70">2</td>
11      <td width="70">3</td>
12      <td width="70">4</td>
13    </tr>
14    <tr>
15      <td height="50">5</td>
16      <td>6</td>
17      <td>7</td>
18      <td>8</td>
19    </tr>
20    <tr>
21      <td height="50">9</td>
22      <td>10</td>
23      <td>11</td>
24      <td>12</td>
25    </tr>
```

```
26      </table>
27      <p><h3>合并表格单元格后</h3></p>
28      <table width="305" height="156" border="1">
29      <tr>
30        <td height="50" colspan="2">1&amp;2</td>
31          <td width="70">3</td>
32        <td width="70" rowspan="3">4&amp;8&amp;12</td>
33      </tr>
34      <tr>
35        <td height="50" width="70">5</td>
36        <td width="70">6</td>
37          <td>7</td>
38      </tr>
39      <tr>
40        <td height="70">9</td>
41        <td>10</td>
42          <td>11</td>
43      </tr>
44        </table>
45      </body>
46    </html>
```



【运行程序】浏览该页面，结果如图 8.6 所示。



图 8.6　合并表格单元格

【深入学习】留意对比代码中对应页面的不同编号表格的位置。合并单元格，这是一个不错的方法。首先编排好单元格的序号，根据希望合并的某些位置的单元格，删除对应序号的单元格。如要合并 1 号和 2 号，则要先删除 2 号单元格，在代码中表现为第 10 行代码。之后再定义将 1 号和 2 号单元格同行内合并。同样，合并一列中的单元格也是这样。本例中注意第 30 行和第 32 行代码的 colspan 和 rowspan 的用法。

注意：如果合并错误位置的单元格，则 colspan 和 rowspan 不起作用。

8.2.4　表格标题

使用<caption>标签给表格添加标题，这个标签放入在<table>标签中，写法是：

```
<table>
<caption>表格的名字</caption>
…
```

<caption>表示标签添加的标题，默认情况下在表格上方居中的位置，它会根据不同表格的宽度来改变位置。

8.2.5　拆分表格

为了便于将表格定位给 CSS 样式表，有时不希望代码中一直都是<tr>标签，可以使用 thead、tfoot、tbody 来拆分表格。thead 定义了表格的首行，tfoot 定义了表格的尾行。这里有意思的是，如果使用了其中一个，那么全部元素都要用上。而且它们的出现次序是 thead、tfoot、tbody，如以下代码所示：

```
<table >
<thead>
    <tr>
        <td>thead</td>
        <td>thead</td>
    </tr>
</theadt>
<tfoot>
    <tr>
        <td>tfoot</td>
        <td>tfoot</td>
    </tr>
</tfoot>
<tbody>
        <td>tbody</td>
        <td>tbody</td>
</tr>
</tbody>
<tbody>
    <tr>
      <td>tbody</td>
    <td>tbody</td>
</tr>
</tbody>
</table>
```

首先是 thead

然后是 tfoot

最后是 tbody



说明：tbody 可以出现多次，但 thead 和 tfoot 只能使用一次。

使用这种写法还有个优点在于，如果工作中制作某些表格长度超出一页的范畴，当打印表格时，表格的表头和页脚将会被打印在包含表格数据的每张页面上。

8.2.6 设置表格的列

虽然 HTML 页面中表格是按照一行一行这样的概念建立起来的，但是可以使用<colgroup>定义表格列的分组。这个标签常和其他两个标签配合使用，一个是<col>标签，一个是<span>标签。<col>标签表示为表格中一个或多个列定义属性值，是仅包含属性的空元素，只能在表格或 colgroup 中使用此元素。如程序 8.3 中使用的<colgroup>和<col>定义表格的列。

【本节示例参考：资料光盘\第 8 章\8-3　使用<colgroup>和<col>定义表格的列.html】。

【实例 8-3】使用<colgroup>和<col>定义表格的列，其源码展示如下：

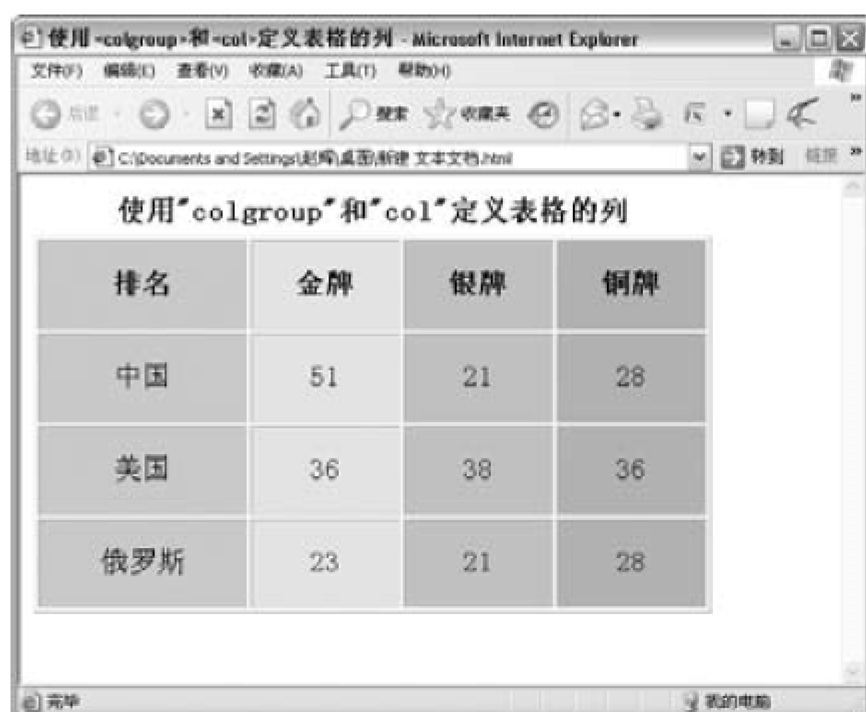
程序 8.3　使用<colgroup>和<col>定义表格的列.html

```
01  <html>
02    <head>
03      <title>使用<colgroup>和<col>定义表格的列</title>
04    <style type="text/css">
05            caption { font-weight:bold; color:#000;
06                       }          /*通过样式表来修饰表格*/
07            .one { background:cyan; text-align: center; }
08            .two { background:ffee22; text-align: center; }
09            .three { background: silver; text-align: center; }
10            .four {    background: #F1B005; text-align: center; }
11    </style>
12  </head>
13    <body>
14      <table width="500" height="280" border="1">
15      <caption>使用"colgroup"和"col"定义表格的列
16      </caption>
17        <colgroup class="one"></colgroup>
18        <colgroup>
19          <col class="two"></col>
20          <col class="three"></col>
21          <col class="four"></col>
22      </colgroup>
23
24  <tr>
25        <th>排名</td>
26        <th>金牌</th>
27        <th>银牌</th>
28        <th>铜牌</th>
29  </tr>
30  <tr>
```

将样式表对应到每个不同部分的列

```
31          <td>中国</td>
32          <td>51</td>
33          <td>21</td>
34          <td>28</td>
35    <tr>
36          <td>美国</td>
37          <td>36</td>
38          <td>38</td>
39          <td>36</td>
40    </tr>
41    <tr>
42          <td>俄罗斯</td>
43          <td>23</td>
44          <td>21</td>
45          <td>28</td>
46          </tr>
47        </table>
48      </body>
49    </html>
```

【运行程序】结果如图 8.7 所示。



排名	金牌	银牌	铜牌
中国	51	21	28
美国	36	38	36
俄罗斯	23	21	28

图 8.7　使用 colgroup 和 col 定义表格的列



注意：这里可以看到，表格标题是在表格上方居中的位置。

【深入学习】这是一个使用 CSS 样式表来改变表格列的背景颜色，当在<style>标签中定义好 4 个样式表之后，把<colgroup>和<col>放在一起，调用这些样式表来改变表格中每一列的颜色。

此外，还可以和 span 属性配合使用，那么代码应该写成：

```
<table>
<colgroup span="3" class="one"
</ colgroup>
```

这种方法不同于前一种的是，它表示前 3 列而不是第 3 列，所以最后在页面中的结果如图 8.8 所示。

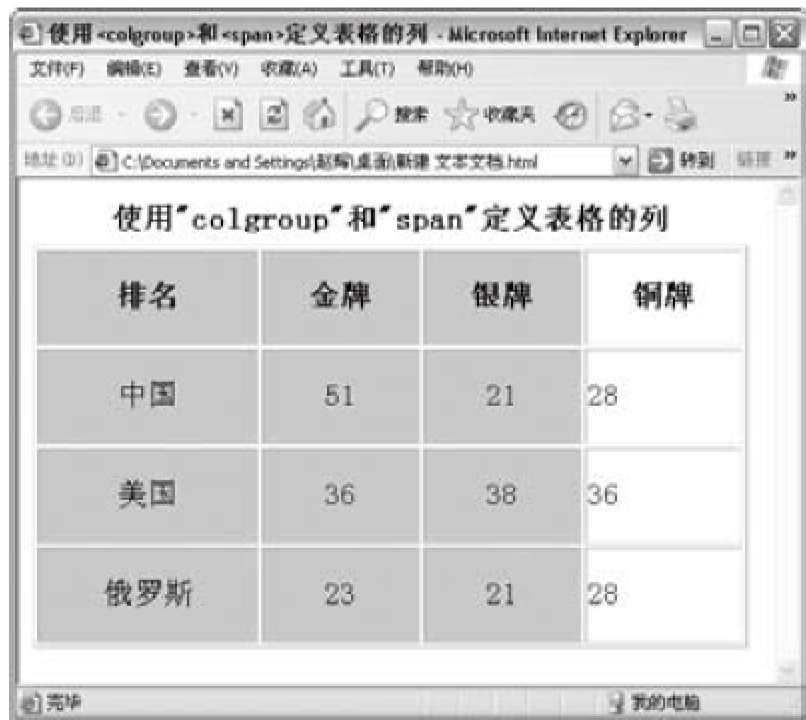


图 8.8　使用 colgroup 和 span 定义表格的列

8.3　修饰单元格

当了解了表格的构建原理之后，进一步讨论的是如何使设计的表格更美观一些。表格是由一个一个的单元格组成的，美化表格的要点就在于如何去美化这些单元格。谈到修饰，最好的方法还是依赖使用样式表。设计者可以利用很多优秀的属性彻底改变表格的样式。

8.3.1　通过 CSS 修饰单元格的边框

修改边框可以使用 border 属性，其不仅可以修改边框的粗细，也能修改边框的颜色和样式。默认情况下，边框的值是 0，即没有边框。边框颜色和文本颜色默认情况下是相同的。一个标准的边框定义在样式表中可以写成这样：

```
border:2px solid red;
```

表示边框的宽度是 2px，红色实体的线。具体如何使用到表格中，如程序 8.4 所示。

【本节示例参考：资料光盘\第 8 章\8-4　修改表格的边框.html】

【实例 8-4】修改表格的边框，其源码展示如下：

程序 8.4　修改表格的边框.html

```
01    <html>
02      <head>
03        <title>修改表格的边框</title>
04      <style type="text/css">
05          table {border:5px blue;                    //设置表格边框样式
06                 }
07          .dotted {border: 3px dotted;               //点状的边框
08                   }
09           .dashed {border: 3px dashed;              //断断续续的边框
```

```
10                  }
11          .double {border: 3px double;          //双线的边框
12                  }
13           groove {border: 3px groove;          //外阴影的边框
14                  }
15               </style>
16           </head>
17           <body>
18              <table width="462" height="242" border="2">
19                  <tr>
20                      <td class="dotted">dotted</td>
21                      <td class="dashed">dashed</td>
22                  </tr>
23                  <tr>
24                      <td class="double">double</td>
25                      <td class="groove">groove</td>
26                  </tr>
27           </body>
28         </html>
```

【运行程序】本例中介绍了如何使用样式表作用于表格的边框，同时也展示了不同边框的样式粗细颜色，结果如图 8.9 所示。



图 8.9　表格边框的样式



说明：为了节约页面，这个例子中只列出了其中 4 种边框的代码写法，而图中 8 种样式边框的源码可以在资料盘中查找到。

【深入学习】此外，设计者还可以分别针对一个单元格的每一条边进行定义，如果这样写：

```
{ border-left:2px dotted; }
```

那么在浏览器中，这个声明值针对于左边的边框。依此类推，还有 border-top、border-right 和 border-bottom，它们分别表示单元格的上边、右边和底边。

8.3.2 合并相邻单元格

<table>标签制定的表格会在所有的单元格之外，再框上一个“四边形”，而每一个单元格又是独立存在的。所以单元格和单元格之间总像是有一条缝隙，有一种方式可以消除这条缝隙，即使用“边框挤压”的属性，如下代码所示：

```
{ border-collapse: collapse;}
```

如果将它运用于表格中，如程序 8.5 所示。

【本节示例参考：资料光盘\第 8 章\8-5　合并单元格之间的边框.html】

【实例 8-5】合并单元格之间的边框，其源码展示如下：

程序 8.5　合并单元格之间的边框.html

```
01   <head>
02      <title>修改表格的边框</title>
03      <style type="text/css">
04            <caption>合并单元格之间的边框
05            </caption>
06       table.one   {border-collapse: collapse;
07                    border:#0099CC
08                    }
09      </style>
10      </head>
11      <body>
12          <table width="191" height="129" border="2" class="one">
13                 <tr bordercolor="#0099CC">
14                       <td width="87">Hui</td>
15                       <td width="86">Bobo</td>
16                </tr>
17                 <tr bordercolor="#0099CC">
18                       <td>Huiji</td>
19                       <td>Jhn</td>
20               </tr>
21      </table>
22   </body>
```

【运行程序】结果如图 8.10 所示。

【深入学习】这样，单元格和单元格之间的缝隙就消除了。此外，“边框挤压”中还可以写成“{border-collapse: separate;}”，表示为单元格之间分离。此外，还有一个 border-spacing 属性具有类似功能，使用它可以控制单元格之间的距离，但是很可惜，这一属性并不能被一些主流浏览器接受，如 IE、Firefox 浏览器。

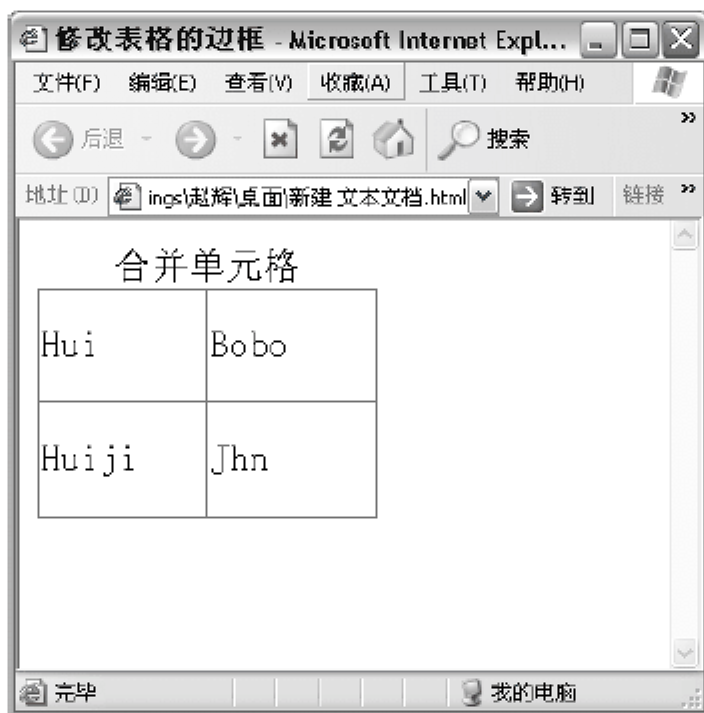


图 8.10　合并单元格之间的边框

8.4　编辑单元格中的内容

表格是由许多个单元格组成的，而这些单元格中又可以放入多种类型的页面内容，如文本、图像或超链接等，甚至可以再放入新的表格。这种表格的嵌套曾经是非常流行的布局页面的方法，只是这种方法太过繁琐。通过样式表来修饰表格中的内容就容易多了。

8.4.1　单元格中文本与单元格大小

单元格的大小会随着格内文本的长度自行缩放。虽然通过数值可以固定单元格的大小，但是如果文本的长度超过所设置的单元格长度，那么依然会根据文本的长度来做决定。使用 table-layout 属性可以限制单元格随文本长度而改变，如下代码所示：

```
{table-layout : fixed;}
```

这样带来的好处是浏览器可以更快地渲染出表格。因为当浏览器解析到表格的第一行时，就能确定所有单元格的大小。使用时，需要在代码中给出确定的长度数值，如第一行的 td 或 th，如程序 8.6 所示为如何限制单元格的大小。

【本节示例参考：资料光盘\第 8 章\8-6　限制单元格大小.html】

【实例 8-6】限制单元格大小，其源码展示如下：

程序 8.6　限制单元格大小.html

```
01   <html>
02     <head>
03       <title>限制单元格大小</title>
04     <style type="text/css">
05     table   { table-layout : fixed;                    //表格单元格的限制
06               }
07     </style>
08     </head>
```

```
09     <body>
10     <table border="2">
11       <tr>
12         <td width="100" >页面文本 1</td>
13         <td width="100" >页面文本 2</td>
14       </tr>
15       <tr>
16         <td >博客社博客社博客社博客社博客社</td>
17         <td ></td>
18       </tr>
19     </body>
20   </html>
```

【运行程序】浏览该页面，结果如图 8.11 所示。

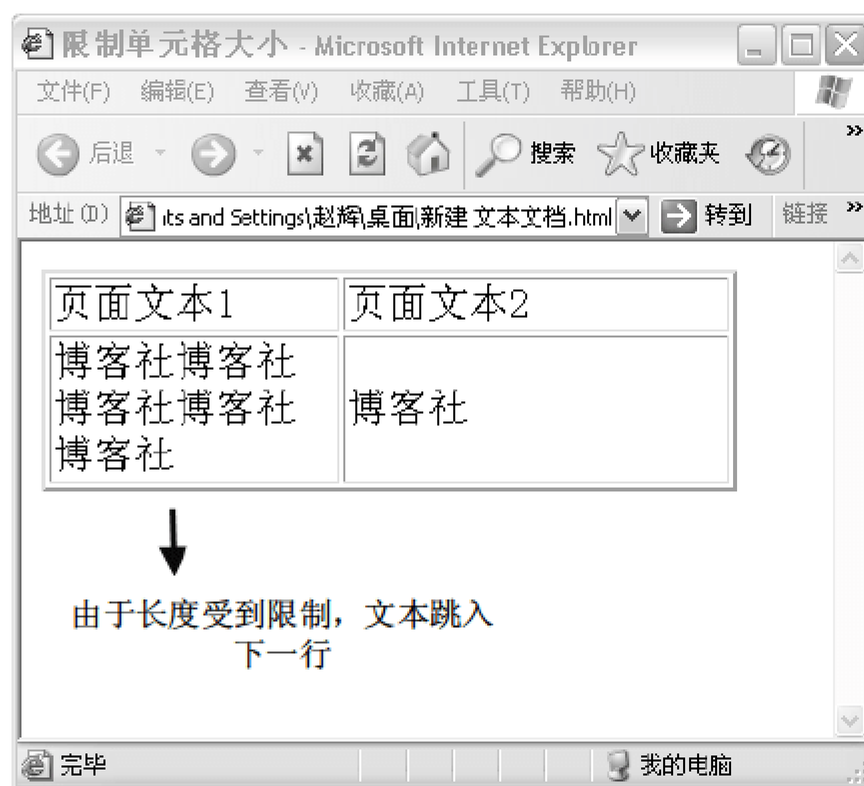


图 8.11　限制单元格大小



说明：有些浏览器不能很好地支持 table-layout 属性，所以使用时要留心。

8.4.2　修饰单元格中的内容

通过 CSS 定义单元格中的文本时，可以专门地指定给某一行、某一列，或者是整个表格，如文本颜色、背景、背景图片等。例如：

```
td { text-align:center;
     font:.7em  幼圆;
     color:#334542;
     background-color:#ddd;
     }
```

样式表中可放入的属性有很多，也有一些是专门属于表格的样式表，如表 8.1 所示，有兴趣的读者可以尝试一下效果。

表 8.1　修饰表格的一些属性

属　性	作　用
border	所有4个边的属性设置在一个声明中
border-style	设置元素所有边框的样式，或者单独地为各边设置边框样式
border-color	设置边框中可见部分的颜色，或为 4 个边分别设置颜色
border-width	为元素的所有边框设置宽度，或者单独地为各边框设置宽度
border-bottom	针对于下边框的所有属性设置到一个声明中
border-bottom-color	设置元素的下边框颜色
border-bottom-style	设置元素的下边框样式
border-bottom-width	设置元素的下边框宽度
border-left	针对于左边框的所有属性设置到一个声明中
border-left-color	设置元素的左边框颜色
border-left-style	设置元素的左边框样式
border-left-width	设置元素的左边框宽度
border-right	针对于右边框的所有属性设置到一个声明中
border-right-color	设置元素的右边框颜色
border-right-style	设置元素的右边框样式
border-right-width	设置元素的右边框宽度
border-top	针对于上边框的所有属性设置到一个声明中
border-top-color	设置元素的上边框颜色
border-top-style	设置元素的上边框样式
border-top-width	设置元素的上边框宽度
border-collapse	设置是否把表格边框合并为单一的边框
border-spacing	设置分隔单元格边框的距离（仅用于separated borders模型）
caption-side	设置表格标题的位置
empty-cells	设置是否显示表格中的空单元格（仅用于separated borders模型）
table-layout	设置显示单元、行和列的算法

8.5　案例：制作球赛积分表

足球是一项如此有魅力的比赛，每当大赛来临，都能吸引一大批狂热的球迷，每每看到大赛的时间表、积分表都显得特别有活力。下面的例子中将介绍如何用表格制作出一个鲜亮的球赛积分表，如程序 8.7 所示。可以把它放在个人博客上或是个人网站内，借助它和其他球友们一起享受球迷的生活。

【本节示例参考：资料光盘\第 8 章\8-7　制作球赛积分表.html】

【实例 8-7】制作球赛积分表，其源码展示如下：

程序 8.7　制作球赛积分表.html

```
001  <html>
002  <head>
003  <title>制作球赛积分表</title>
004  <style type="text/css">
005  #messi {
006            width:700px;
007        border-collapse:collapse;
008        font-family:微软雅黑;
009        text-align:center;
010        margin:0px auto;
011  }
012  #messi caption {
013        border-weight:bold;
014        padding:6px 0px;
015        color:#3D580B;
016        font-size:25px;
017                        }
018  #messi th,td {
019        border:1px solid #aaa;
020                   }
021  #messi thead th {
022        border-bottom:2px solid #3D580B;
023        background-color:#8FC629;
024        color:#fff;
025        padding:5px 0px;
026                    }
027  #messi th.title {
028        background-color:#E3E685;
029                    }
030  #messi th {
031        background-color:#F2F4B9;
032               }
033  #messi tfoot td {
034        border-width:0px;
035        text-align:right;
036        font-size:12px;
037        color:#777;
038                     }
039
040  #kaka th.title {
041        background-color:#ffd56c;
042                   }
043  #kaka th {
```

以 messi 命名的样式表定义了“西甲”部分的表格样式

以 kaka 命名的样式表定义了“意甲”部分的表格样式

```
044         background-color:#ffe8ae;
045                 }
046  #cr th.title {
047         background-color:#FFCC33;
048                 }
049  #cr th {
050         background-color:#fff255;
051           }
052  </style>
053  </head>
054
055  <body>
056  <table id="messi">
057  <caption>欧洲豪门最新积分榜(截止到 2008-09-14)</caption>
058  <thead>
059    <tr>
060      <th>球队</th>
061      <th>胜   </th>
062      <th>平   </th>
063      <th>负   </th>
064      <th>积分</th>
065      <th>排名</th>
066    </tr>
067  </thead>
068  <tfoot>
069    <tr>
070      <td colspan="6">zhaohui </td>
071    </tr>
072  </tfoot>
073  <tbody >
074    <tr>
075      <th colspan="6" class="title">西班牙甲级联赛</th>
076    </tr>
077    <tr>
078      <th>皇家马德里</th>
079      <td>0</td>
080      <td>0</td>
081      <td>1</td>
082      <td>0</td>
083      <td>15</td>
084    </tr>
085    <tr>
086      <th>巴塞罗那</th>
087      <td>0</td>
088      <td>1</td>
089      <td>1</td>
```

以 cr 命名的样式表定义了“英超”部分的表格样式

```
090        <td>1</td>
091        <td>14</td>
092      </tr>
093      <tr>
094        <th>瓦伦西亚</th>
095        <td>1</td>
096        <td>0</td>
097        <td>0</td>
098        <td>3</td>
099        <td>3</td>
100      </tr>
101  </tbody>
102   ......
166  </tbody>
167       </table>
168     </body>
169  </html>
```



说明：由于第 102～165 行中间代码部分与前面部分类似，故省略以节约页面，可参考光盘中源码。

【运行程序】其中，第 1～54 行代码是设定好的 CSS 样式表的头部，id 命名为 messi、kaka 和 cr 的样式表分别代表了“西甲”、“意甲”和“英超”这 3 部分表格。在它们各自的扩展下，又定义了一系列针对表头、单元格的样式表。第 55～169 行代码是将样式表应用于表格中的不同内容中，结果如图 8.12 所示。

图 8.12　制作球赛积分表

【深入学习】这个例子中，要注意样式表是绑定在哪些对象上的，如代码第 21～26 行，作用于图中“球队、胜、平...”这一行。第 18～20 行中的代码“#messi th,td”，定义了表格中所有边框的样式。注意在这个声明中是一个逗号，而非点号。第 75 行代码使用 colspan 属性设置了“西班牙甲级联赛”文本的样式，类似这样的用法还有第 104 行和第 132 行代码。



说明：本例中出现了 margin 和 padding 属性，它们属于“框”布局的属性，可参照第 10 章。

8.6　小　　结

本章介绍了表格的用法，可以发现，设置表格的方法不难，但是表格缺少一些固定的规律，编辑的方式多而灵活。由于不同浏览器支持的效果大不相同，所以表格一直是比较难控制的一样东西。好在现在设计者不用再考虑用表格来布局页面了，虽然偶然也会使用，但是更多的时候，表格只是用来制作表格。本章主要的知识点有：

- 构建表格的方式。
- 控制表格的行高宽度，以及单元格。
- 使用<caption>设置表格的标题。
- 为了更方便应用 CSS，将表格拆分为表头、表尾和表体。
- 合并单元格，改变单元格边框样式。
- 控制表格的列。
- 编辑修饰单元格中的内容。
- 通过实例学习如何综合运用 CSS 样式表来制作一个表格。

表格的用途很广，是页面中最常见的内容之一，希望读者多加练习。

第 9 章　创建框架结构的页面

框架指的是页面的一种布局，这种布局和之前提到的“表格布局”以及之后将学习的“CSS+DIV”布局都是不同的。那么具体框架布局是怎样的呢？事实上，框架布局也许缺少了那么一些灵活性，但它的特点是可以将浏览器分割成几部分，这是其他标签无法做到的，本章将介绍的便是这种技能。本章主要的知识点如下。

- 学习如何创建框架集。
- 掌握基本的页面布局。
- 通过一些属性修饰框架。
- 学习如何在框架中实现页面链接以及锚点链接。
- 创建浮动框架。

9.1　创建窗口框架页面

有这样一种网页，导航栏放在左侧，右侧是页面主体，当浏览者单击左侧导航栏时，右侧的页面会刷新，但是左侧导航栏部分依然保持不变。如图 9.1 所示是一个常见的论坛导航页面。

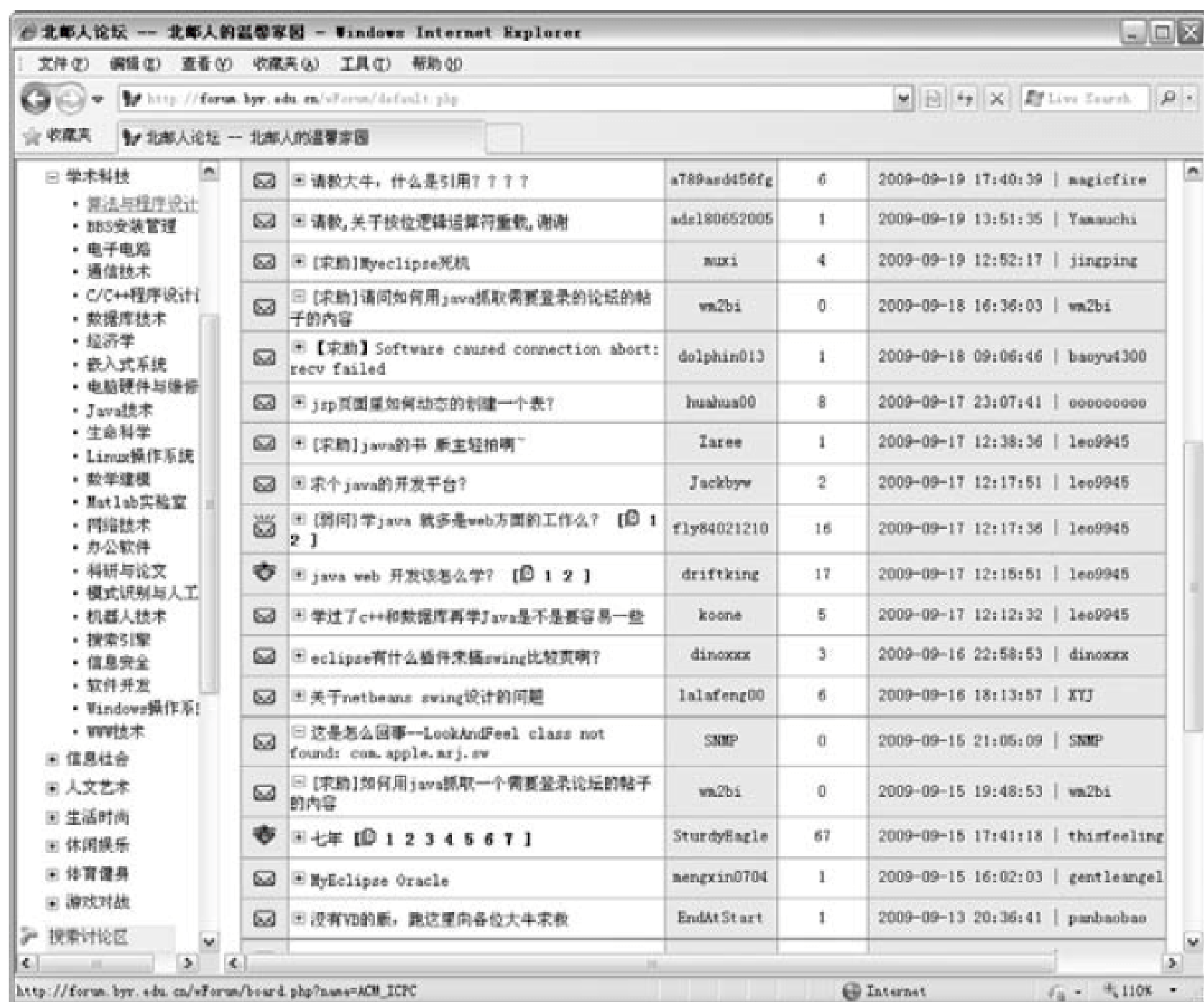


图 9.1　使用框架的页面

类似这样形式的论坛非常多，左侧是导航栏，右侧是论坛主体。单击左侧导航栏，则在网页的右侧显示链接页面，这个布局是将浏览器分为左右两部分。使用者可以在导航栏中查找目录，在右侧的主页中刷新不同的页面。

9.1.1　创建窗口框架的<frameset>和<frame>标签

使用<frame>框架在 HTML 页面中设置框架。那么，当一个浏览器被分成很多个框架时，这些框架放在一起，即称之为框架集。框架集的 HTML 标签为<frameset>，也称之为框架结构标签。所以，如果要在框架中放入内容，采用的方式是通过引用所放内容的路径来加载对象。它的代码如下所示：

```
<frameset ......>              //表明这是一个框架集
    <frame src=...>           //这是其中一个框架中的页面路径
    <frame src=...>
...
</frameset>
```

需要注意的是，不能将<frameset>标签和<body>标签一起使用。原理上，因为框架集分割的是浏览器，也就是说，框架集至少也需要由两个框架组成。所以，不存在只有一个框架的页面。而且，框架集的作用是将多个页面同时展示在浏览器中，同样，也不存在包含框架的独立页面。因此，<frameset>标签和<frame>标签是不能放在<body>标签内的，这样做没有意义。

9.1.2　横向分割窗口

窗口的分割只有横纵两个方向，没有斜方向的分割方法。横向分割窗口，使用 rows 属性。代码如下所示：

```
<frameset rows="框架高度,框架高度,...*">
```

- 框架高度：可以使用百分比或者像素来表示。
- *：表示最后一个框架的高度，即剩下来的最后一个，也就无须再用数值表示。

这里通过一个例子来说明如何实现使用框架横向分割窗口。例如，程序 9.1 中横向分割窗口的页面代码。

【本节示例参考：资料光盘\第 9 章\9-1　横向分割窗口.html】

【实例 9-1】横向分割窗口的方法，其源码展示如下：

程序 9.1　横向分割窗口.html

```
01   <html >
02     <head>
03     <title>横向分割窗口</title>
04     </head>
05           <--以下部分是框架集-->
06     <frameset rows="40%,40%,*" >      //划分不同大小的框架
07       <frame ></frame >
08       <frame ></frame >
```

```
09          <frame ></frame >
10       </frameset>
11    </html>>
```

【运行程序】最终页面结果如图 9.2 所示。

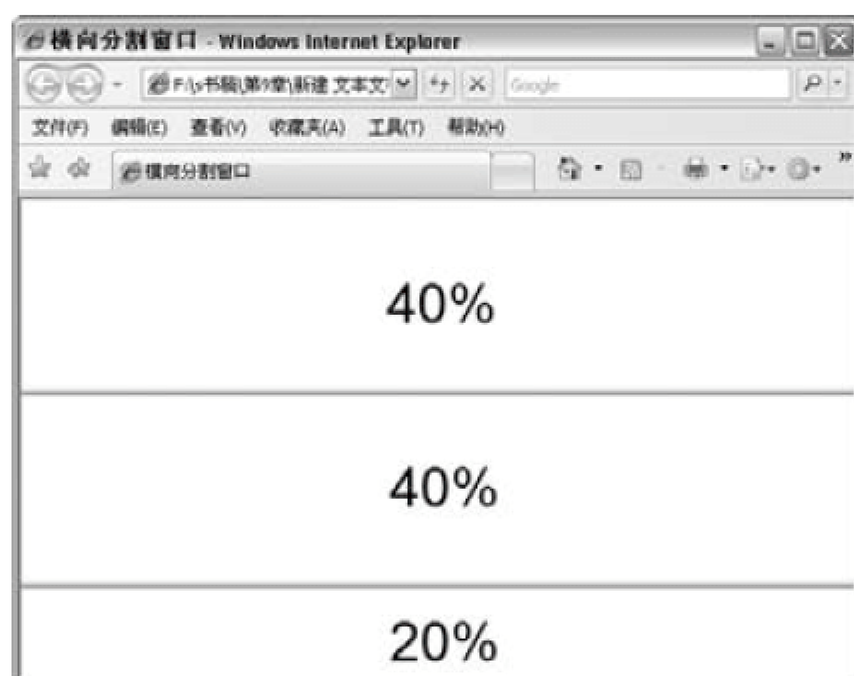


图 9.2　横向分割窗口



注意：代码“<frameset rows="40%,40%,*" >”最后位的星号符，浏览器会默认为是剩下的框架区域，在此例中即为 20%。

可以看出，最下面的一栏被默认定义为 20%，事实上，这也是设计者希望看到的结果。

9.1.3　纵向分割窗口

除去横向分割窗口，另一种形式则是纵向分割窗口。当纵向分割窗口时，使用的是 cols 属性。同样，用百分比或者像素都可以来表示框架横向的宽度。如程序 9.2 所示的纵向分割窗口。

【本节示例参考：资料光盘\第 9 章\9-2　纵向分割窗口.html】

【实例 9-2】纵向分割窗口的方法，其源码展示如下：

程序 9.2　纵向分割窗口.html

```
01    <html >
02       <head>
03       <title>纵向分割窗口</title>
04       </head>
05             <--以下部分是框架集-->
06       <frameset cols="20%,40%,*" >          //划分不同大小的框架
07           <frame ></frame >
08           <frame ></frame >
09          <frame ></frame >
10       </frameset>
11    </html>>
```

【运行程序】结果如图 9.3 所示。

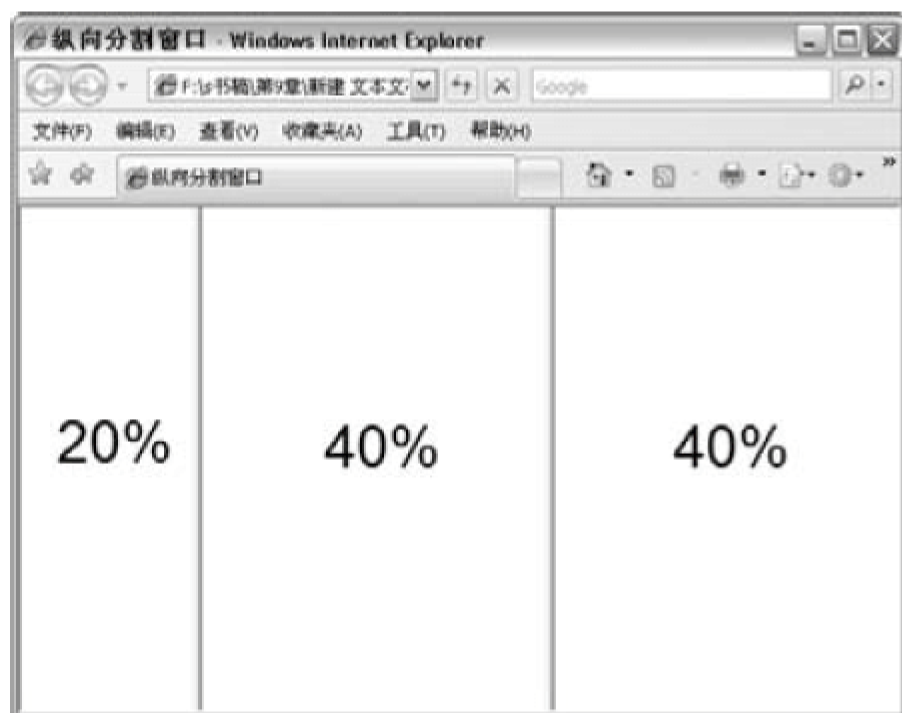


图 9.3　纵向分割窗口



注意：在 HTML 中，<frame>标签不需要关闭，但是在 XHTML 中，<frame>标签必须被正确地关闭。

9.1.4　框架的嵌套

框架嵌套就是说如果同时混合使用横纵结构。即在分隔号的框架中进行新的窗口分割，之后将这个框架替换成一个新的框架集。那么，写法上只要在<frameset>标签中再放入<frameset>标签即可。如程序 9.3 所示是一个简单的框架嵌套的页面代码。

【本节示例参考：资料光盘\第 9 章\9-3　框架的嵌套.html】

【实例 9-3】框架嵌套的方法，其源码展示如下：

程序 9.3　框架的嵌套.html

```
01  <html >
02      <head>
03          <title>框架的嵌套</title>
04      </head>
05        <!--以下部分是框架集-->
06      <frameset cols="25%,*%">
07                  <frame>
08                  <frameset rows="40%,*%">                //划分不同大小的框架
09                          <frame>
10                          <frame>
11                      </frameset>
12          </frameset>
13  </html>>
```

【运行程序】结果如图 9.4 所示。

【深入学习】代码第 8～11 行，是一个新的框架集。同时，这个框架集可以看成是大框架下的一个子框架。这是一个先纵向分割，接着在子框架中横向分割的案例。如果代码第 6 行和第 8 行交换一下，则效果显示如图 9.5 所示。



图 9.4　嵌套的框架布局一



图 9.5　嵌套的框架布局二

图 9.5 所示为先横向分割，接着在子框架中纵向分割窗口的效果。所以页面发生了变化。这是两种最常见的框架布局结构，第一种是左右布局的页面，第二种则是上下布局的页面。异曲同工，从这个改变中可以看到页面设计的魅力。

9.1.5　将页面放入窗口框架中

将页面放入窗口框架中，需要在每一个框架中通过路径的方式来添加页面。如图 9.5 所示，若一个框架集中包含 3 个框架，那意味着要准备 3 个页面分别放入相应的框架中。这里通过一个例子来说明如何将页面加入框架集。准备 3 个页面 red.html、white.html 和 blue.html。具体的方法如程序 9.4 所示。

【本节示例参考：资料光盘\第 9 章\9-4　将页面放入到窗口框架中.html】



说明：每一个框架对应着一个页面。所以，如果窗口中划分太多框架，也是一种不妥当的设计方式。

【实例 9-4】将页面放入到窗口框架中，其源码展示如下：

程序 9.4　将页面放入到窗口框架中.html

```
01  <html >
02      <head>
03          <title>将页面放入到窗口框架中</title>
04      </head>
05          <frameset rows="40%,*%">                      //这是框架集
06                  <frame src="red.html">                //通过路径，引入 red 页面
07                      <frameset cols="38.2%,*%">        //这是子框架集
08                          <frame src="复件 white.html">  //通过路径，引入 white 页面
09                          <frame src="blue.html">       //通过路径，引入 blue 页面
```

```
10                  </frameset>
11          </frameset>
12    </html>>
```

【运行程序】结果如图 9.6 所示。



图 9.6　将页面放入窗口框架中

【深入学习】页面中通过 src 属性告诉框架与其相对应的页面，并且将页面添加在其中。这是一个上下分割的页面，上部分框架中是背景为红色的页面。而下部分框架集中，分割成左右框架。左边是背景为白色的页面，右边则是背景为蓝色的页面。

9.2　花点心思修饰框架的细节

设计页面没有太多复杂的技术，但是要设计出优秀的页面，关键在于对于细节的把握，也就是所谓的“细节出大师”。在设计这一领域中细节是一条重要的准则。页面是给浏览互联网的用户提供的，因此，必须要考虑页面给使用者的友好度。一个具有亲和力的页面会让浏览者对这个页面留下深刻的印象。

9.2.1　给无法处理框架的浏览器添加注释说明

在进行框架设计的页面时，会遇到不能显示框架的浏览器。在这种情况下，可以使用<noframe>标签加以注释。如程序 9.5 所使用的注释方法。

【本节示例参考：资料光盘\第 9 章\9-5　使用<noframe>标签注释.html】

【实例 9-5】使用<noframe>标签注释，其源码展示如下：

程序 9.5　使用<noframe>标签注释.html

```
01    <html>
02      <frameset cols="20%,30%,*">              //划分不同大小的框架
```

```
03          <frame >
04          <frame >
05          <frame >
06           <noframes>                              //用来声明当浏览器不能处理框架时的情况
07              <body>很抱歉，您的浏览器无法处理框架！
08              </body>
09           </noframes>
10      </frameset>
11   </html>
```

【深入学习】当使用了第 6～10 行这样的声明，表示如果浏览器无法处理框架，则<noframes>标签开始起作用。那么此时浏览器将显示<body>标签的内容。告诉使用者：“很抱歉，您的浏览器无法处理框架！”



说明：在目前的浏览器中，大部分都能正确处理框架。

9.2.2 固定框架的位置

在<frameset>标签的框架集中，虽然框架的位置是按照事先设定好的出现在浏览器中，但是框架的边框并非是固定的，如果浏览者拖拽框架集的边框，框架的大小是可以随意改变的。若设计者希望固定框架的尺寸，可以使用<noresize>标签来定位边框的位置。如程序 9.6 的页面，它的框架是固定的。

【本节示例参考：资料光盘\第 9 章\9-6　使用<noresize>标签固定框架的位置.html】

【实例 9-6】使用<noresize>标签固定框架的位置，其源码展示如下：

程序 9.6　使用<noresize>标签固定框架的位置.html

```
01   <html>
02      <frameset rows="20%,30%,*">                          //划分框架
03           <frame noresize="noresize" >                    //固定框架的位置
04           <frame noresize="noresize" >
05              <frameset cols="35%,25%,*">
06                    <frame noresize="noresize" src="red.html">
07                    <frame noresize="noresize" src="blue.html">
08              </frameset>
09      </frameset>
10   </html>
```

这样设定之后，边框的位置就被固定住了，因此，框架的尺寸不会被随意地改变。这样的设定，可以使页面显得更工整。

9.2.3 框架中设置滚动条

窗口框架中有个细节，当页面中的内容超出框架的范围时框架的底边会出现滚动条。这个滚动条

也是可以设置的，通过 scrolling 属性可以实现这种控制。实现它的方法如实例 9-7 所示。

【本节示例参考：资料光盘\第 9 章\9-7　设置滚动条.html】

【实例 9-7】设置滚动条的方法，其源码展示如下：

程序 9.7　设置滚动条.html

```
01    <html >
02        <head>
03            <title>设置滚动条</title>
04        </head>
05          <!—以下部分是框架集-->
06        <frameset rows="80%,*%">
07                <frame   src="制定积分表.html" scrolling=auto> //设置滚动条是自动的
08                <frameset cols="38.2%,*%">
09                        <frame src="white.html" scrolling=no >   //设置滚动条是不显示的
10                        <frame src="blue.html" scrolling=yes >   //设置滚动条是显示的
11                    </frameset>
12            </frameset>
13    </html>>
```

【运行程序】如图 9.7 所示的页面中标注线框的便是滚动条。这是已经设置好的滚动条样式。



图 9.7　框架中的滚动条

【深入学习】代码第 7、9、10 行的 scrolling 属性分别定义为 auto、no 和 yes，它们分别表明不同的作用。

- ❑ auto：根据页面的内容是否超出框架范围，而决定是否出现滚动条。如图 9.7 中顶部框架中页面超出了框架范围，右侧和边框底部都出现了滚动条。
- ❑ no：表示为不出现滚动条。
- ❑ yes：表示为无论页面内容是否超出框架范围，都将显示滚动条。如图 9.7 中页面的下方右侧，表现为灰色的滚动条。

9.3　修改框架边框的样式

框架的边框也是可以修改的，通过一些简单的属性修饰，可以改变框架边框的表现形式。如边框的粗细、颜色，或者是边框的边距。使用这些特点，可以作出一些有意思的页面效果。

9.3.1　判定边框是否显示

在有些情况下，灰色的边框会很麻烦，看上去像把页面剖解得支离破碎。使用 frameborder 属性可以决定是否显示边框。它的写法是：

```
<frame frameborder="0" src="">
```

0 表示不显示边框，如果写成 1，则是显示边框。如图 9.8 所示为程序 9.8 定义的不显示边框的页面。

【本节示例参考：资料光盘\第 9 章\9-8　判定边框是否显示.html】

【实例 9-8】判定边框是否显示的方法，其源码展示如下：

程序 9.8　判定边框是否显示.html

```
01   <html>
02       <head>
03           <title>判定边框是否显示</title>
04       </head>
05     <frameset cols="50%,*" border="2px">                        //划分框架大小
06           <frame frameborder="0" src="blue.html">
07           <frame   frameborder="0" src="white.html" >
08     </frameset>
09   </html>
```

【运行程序】浏览该页面，结果如图 9.8 所示。

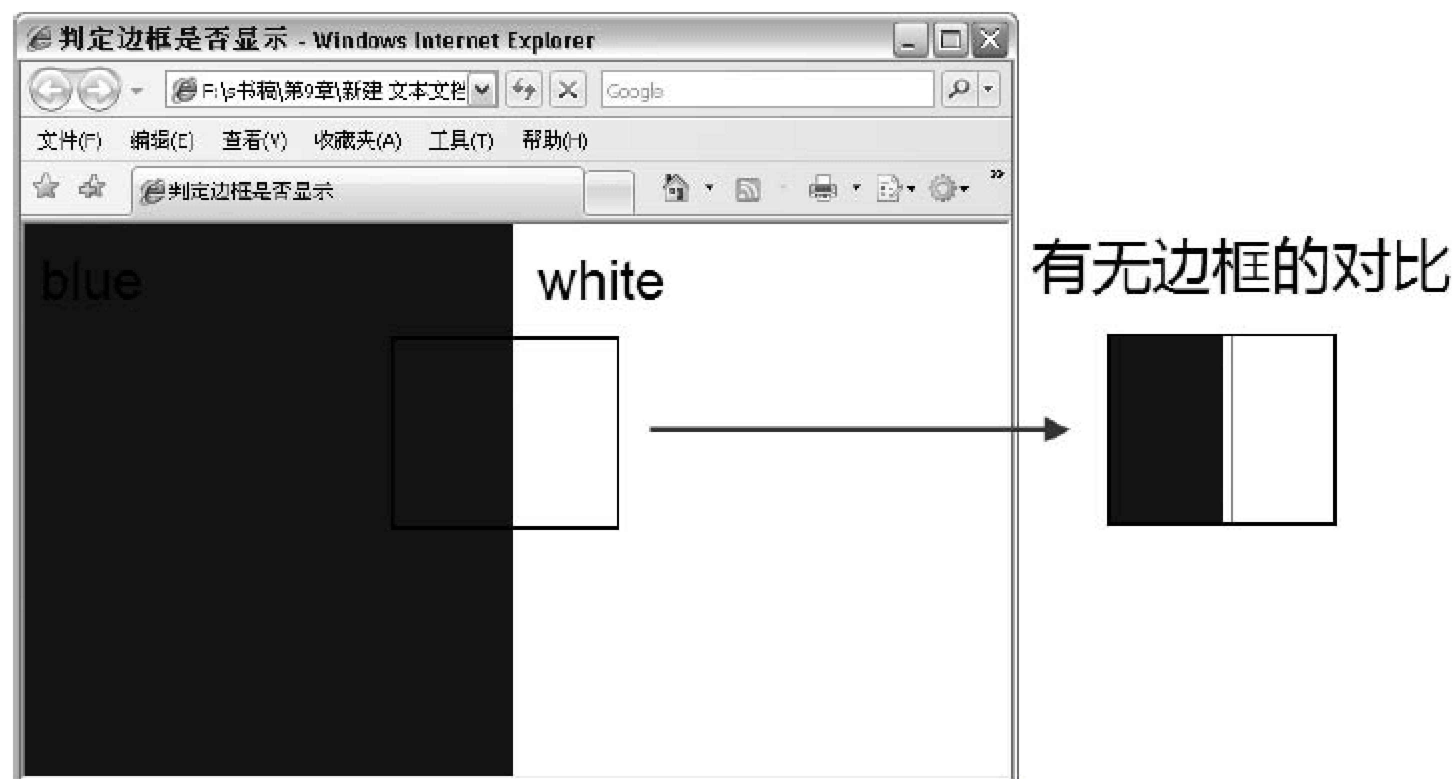


图 9.8　有无边框的对比



注意：不需要边框并不代表边框的宽度为 0，例子中边框的宽度是 2px，最好的方法是同时设定边框的宽度为 0。

9.3.2　改变边框的表现效果

border 表示框架的边框，在这个属性上可以扩展出一些新的特性，如 bordercolor 表示修改边框的颜色。通过修改 border 扩展的一些属性可以改变边框的表现效果。如图 9.9 所示为程序 9.9 中修改过的边框样式。

【本节示例参考：资料光盘\第 9 章\9-9　改变边框的表现效果.html】

【实例 9-9】 改变边框的表现效果，其源码展示如下：

程序 9.9　改变边框的表现效果.html

```
01    <html >
02       <head>
03          <title>改变边框的表现效果</title>
04       </head>
05          <frameset rows="30%,70%"   border=23px bordercolor="#FF0000">
06                      <!—定义边框宽度为 23px，边框颜色是红色-->
07                     <frame >
08                     <frameset cols="40%,*"   bordercolor="#000000">
09                            <frame>
10                            <frame >
11          </frameset>
12    </html>
```

【运行程序】 浏览该页面，结果如图 9.9 所示。



图 9.9　改变边框的表现效果

【深入学习】 代码中第 5 行和第 8 行，定义了页面中框架的边框的宽度为 23px，分别是红色和黑色的边框。



注意：如果在先前一级，如第 5 行中定义了边框的宽度，那么作为嵌入其中的边框，其边框宽度会跟随上一级。即第 8 行虽然没有定义边框的宽度，但是浏览器会默认为这个边框宽度是 23px。

9.3.3　边框的边距

边距是指框架内页面内容离边框的距离。使用 marginwidth 属性设置左右两边的边距，使用 marginheight 属性设置上下两边的边距。如图 9.10 所示为运行程序 9.10 后的效果。

【本节示例参考：资料光盘\第 9 章\9-10　设置边框的边距.html】

【实例 9-10】设置边框的边距，其源码展示如下：

程序 9.10　设置边框的边距.html

```
01  <html >
02    <head>
03      <title>设置边框的边距</title>
04    </head>
05    <frameset rows="30%,30%,*" >
06      <frame src="red.html" marginwidth=50px >//设置页面内容距离边框的距离是 50px
07      <frame src="blue.html" marginwidth=100px >
08            <frameset cols="40%,*" >
09                 <frame    src="red.html" marginheight=50px>
10                 <frame src="复件 white.html" marginwidth=100px marginheight=50px>
11    </frameset>
12  </html>
```

【运行程序】最终在浏览器中的显示效果如图 9.10 所示。

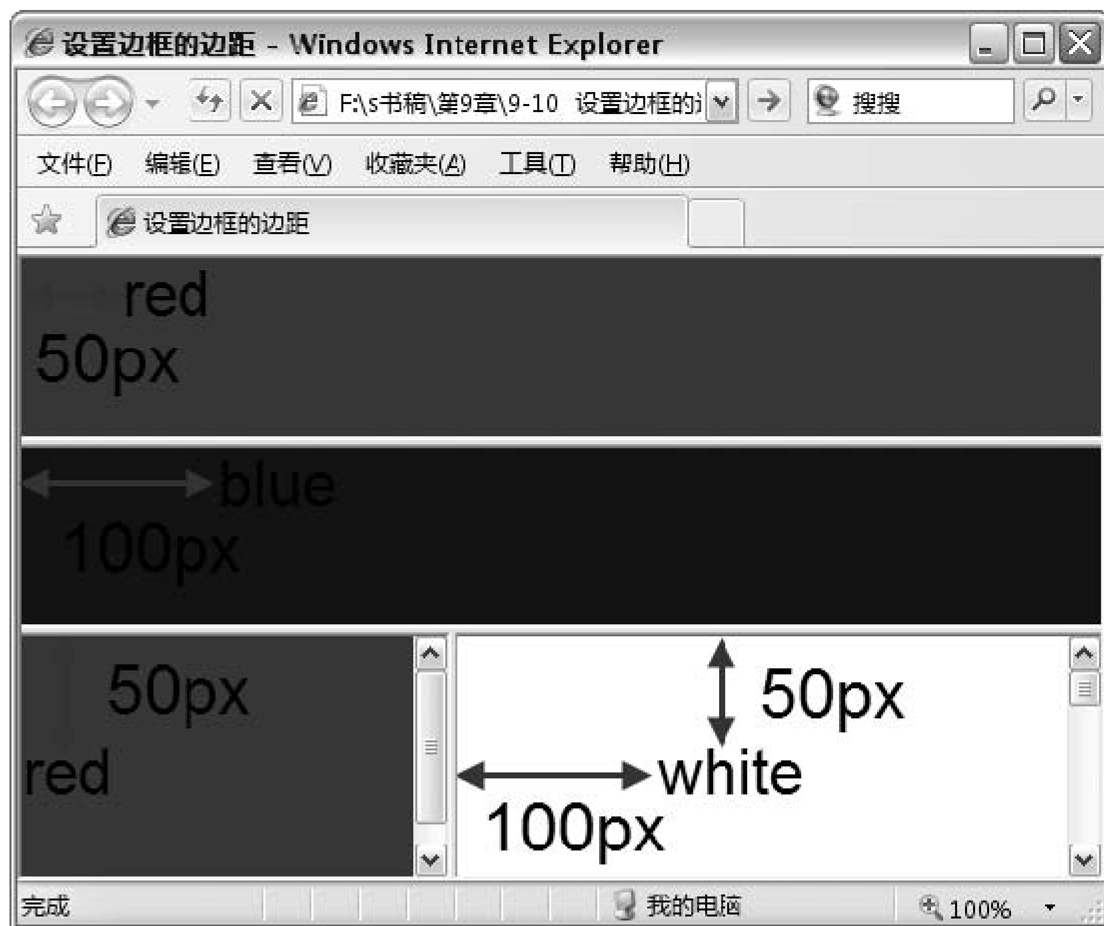


图 9.10　设置边框的边距



说明：marginwidth 和 marginheight 是两个很有实用意义的标签属性，在之后的 CSS+DIV 布局中还会再次遇到。

【深入学习】在本例中可以看到页面中的文本。例如，red、blue 和 white 在不同的框架中，根据不同框架下的 marginwidth 和 marginheight 属性设置，出现在不同的位置。这是一种很好的布局定位的方式。

9.4　框架集中页面之间的链接

本章一开始，提到了论坛形式的页面，以左边为导航栏、右边为页面的主体部分这样框架集下的布局页面。在本节中，将介绍如何使用超链接配合框架的特性，制作出具有特色的页面。

9.4.1　在指定的框架中打开链接

在一张原始的页面如图 9.11 所示中，可以看到一个简单的页面导航是如何在同一个窗口中实现链接的。



图 9.11　初始页面

页面左侧是一个表格，依次写着“第一栏显示蓝色页面”、“第二栏显示红色页面”、“第三栏显示绿色页面”和“第四栏显示橙色页面”。那么，如何才能让左侧导航栏实现这样的功能呢？现在这个浏览器中放入了 5 个页面，当使用者单击页面左侧的导航栏时，其中的条目会链接到右侧不同的位置。程序 9.11 实现了这个框架集。

【本节示例参考：资料光盘\第 9 章\9-11　使用 name 属性给框架命名.html】

【实例 9-11】使用 name 属性给框架命名，其源码展示如下：

程序 9.11　使用name属性给框架命名.html

```
01    <html >
```

```
02      <head>
03        <title>在指定的框架中打开页面</title>
04      </head>
05      <frameset cols="40%,*" >
06          <frame src="white.html" marginwidth=50px   marginheight=20px >
07                  <frameset rows="25%,25%,25%,*" >
08                   <frame   src="silver.html"   marginheight=20px marginwidth=50px
09                    name="blue">              //给不同的框架位置定义相对应的 name
10                   <frame   src="silver.html"   marginheight=20px marginwidth=50px
11                    name="red">
12                   <frame   src="silver.html"   marginheight=20px marginwidth=50px
13                    name="green">
14                   <frame   src="silver.html"   marginheight=20px marginwidth=50px
15                    name="orange">
16    </html>
```

【深入学习】这段代码中的重点是其中的 name 属性。如代码第 8、9 行中表明，这个框架部分命名为 blue。依次可以推论出，代码分别命名了红色、绿色和橙色的框架位置。接下来，只要设法在浏览器左侧令导航栏中的目录链接到所对应名字的框架位置就可以了。程序 9.12 实现了页面对应不同位置的框架。

【本节示例参考：资料光盘\第 9 章\9-12　使用 target 属性定义链接的框架.html】

【实例 9-12】使用 target 属性定义链接的框架，其源码展示如下：

程序 9.12　使用target属性定义链接的框架.html

```
01    <html>
02      <head>
03        <title>使用 target 属性定义链接的框架</title>
04      <style>
05      body {color:black;                                  //文本颜色为黑色
06           font:2em, arial;                               //文本的字体
07           background-color:white;                        //页面的背景
08           }
09      table td {font:1.2em,幼圆;                          //表格中的文本字体
10            line-height:2.5em;}                           //设置列表之间的距离
11      </style>
12      </head>
13      <body>
14         white
15       <p>
16         <table border="0">
17           <tr>
18           <td><a href="blue.html" marginheight=20px marginwidth=50px
19            target="blue">第一栏显示蓝色页面</a></td>     //使用 target 属性定义目标位置
20           </tr>
```

```
21            <tr>
22            <td><a href="red.html" marginheight=20px marginwidth=50px
23                  target="red">第二栏显示红色页面</a></td>
24            </tr>
25            <tr>
26            <td><a href="green.html"   marginheight=20px marginwidth=50px
27                  target="green">第三栏显示绿色页面</a></td>
28            </tr>
29            <tr>
30            <td><a href="orange.html"   marginheight=20px marginwidth=50px
31                  target="orange">第四栏显示橙色页面</a></td>
32            </tr>
33      </body>
34   </html>
```

【运行程序】最终显示的效果如图 9.12 所示。

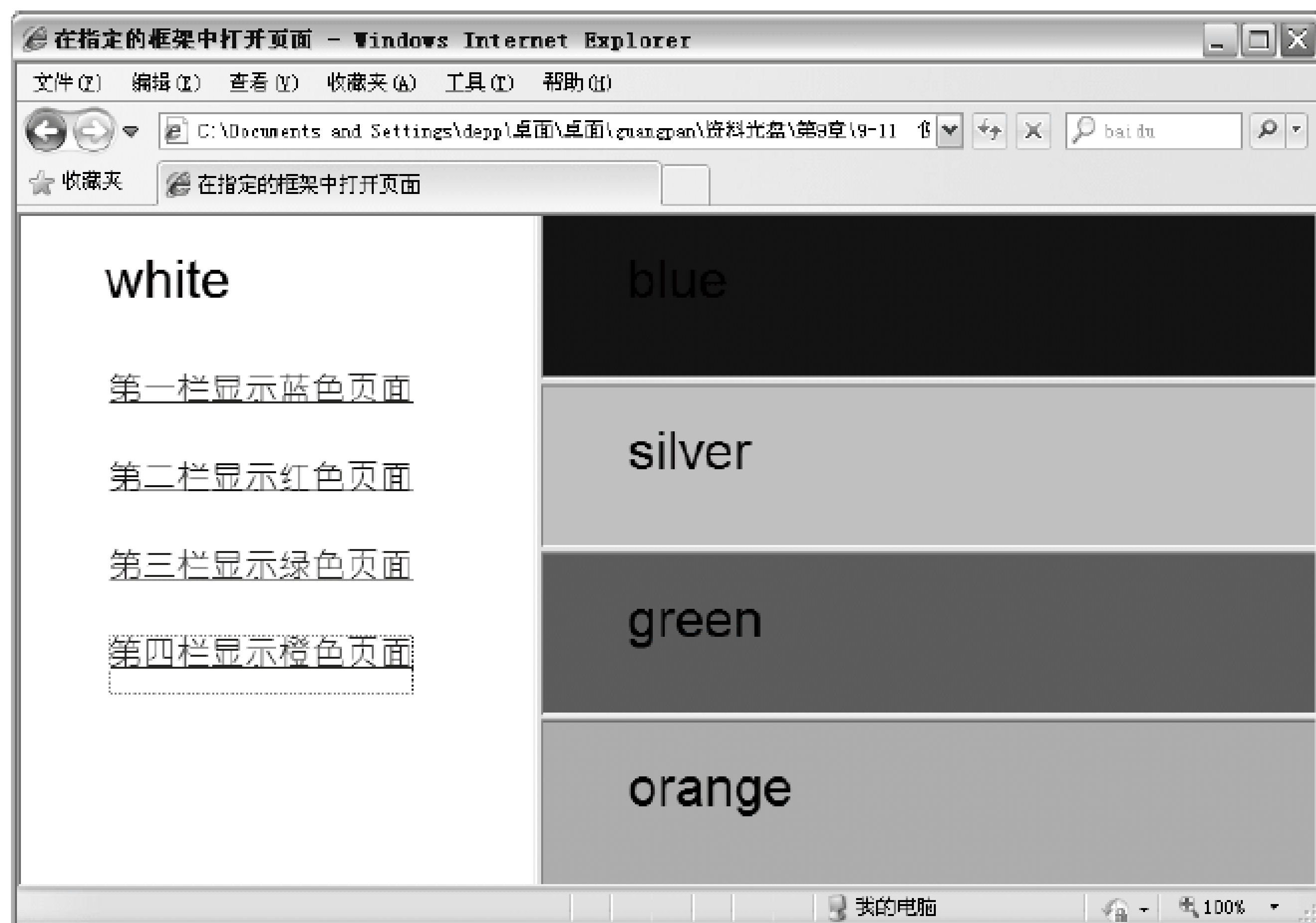


图 9.12　最终页面效果

【深入学习】代码第 18 行中，文本“第一栏显示蓝色页面”超链接到网页 blue.html，而决定放置于哪个页面取决于第 19 行中的“target="blue"”。target 属性使超链接的页面放置于命名为 blue 的框架中，由于在程序 9.10 中已经命名好了 blue 框架，所以，当单击“第一栏显示蓝色页面”超链接时，blue 页面将出现在浏览器右侧的第一栏。同样原理，其他 3 色的页面也能出现在设定好的框架中。



说明：如果单击页面的第二栏，同样会在页面右侧第二栏链接到 red.html。最终页面可参考光盘资料\第 9 章\9-11&9-12 指定框架内实现页面跳转.html。

9.4.2　框架内的锚点链接

使用 name 属性还可以实现在框架内锚点链接。在框架集中设置页面路径的同时指定锚点的位置。

如图 9.13 所示是一张左右分割布局的页面。

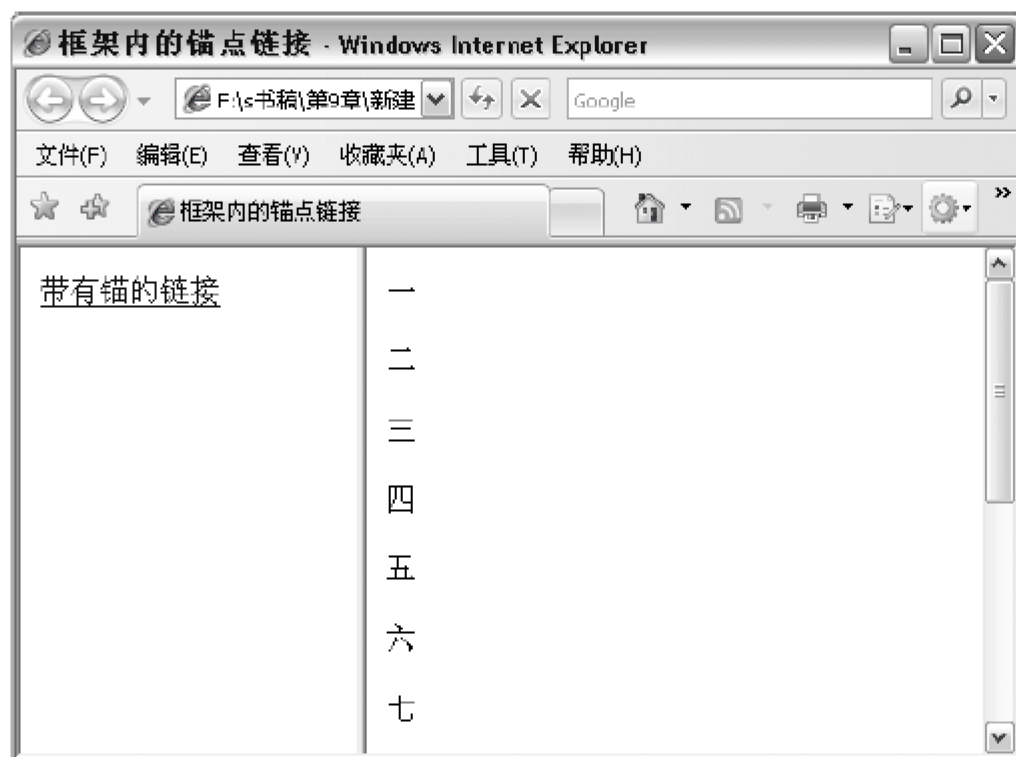


图 9.13 框架内的锚点链接初始页面

在这个页面中，设计者希望当单击左侧边框中文本“带有锚的链接”时，右侧边框中页面跳转到相应的文本位置。那么，在左侧边框中的页面写入页面代码：

```
<a href ="12.html#aaa" target ="showframe">带有锚的链接</a>
```



说明：因为只是一个基本的简单链接页面，这里并没有给出左侧边框中的页面源码，可参考资料光盘。

右侧框架中的页面保存为页面文件 12.html，右侧框架命名为 showframe，所以这句代码的意思表明，链接到 showframe 框架内 12.html 页面的 aaa 位置。因此，右侧框架中的页面代码中需要有这样注明的一句代码：

```
<a name="aaa">十四</a>
```

在文本“十四”的位置定义为“aaa”的锚点。那么，最终框架集的代码如程序 9.13 实现的锚点链接。

【本节示例参考：资料光盘\第 9 章\9-13 框架内的锚点链接.html】

【实例 9-13】框架内的锚点链接，其源码展示如下：

程序 9.13 框架内的锚点链接.html

```
01  <html >
02    <head>
03      <title>框架内的锚点链接</title>
04    </head>
05  <html>
06      <frameset cols="180,*">
07            <frame src="content.html">
08            <frame src="12.html#aaa" name="showframe">
```

```
09        </frameset>
10    </html>
```

【运行程序】浏览该页面，结果如图 9.14 所示。

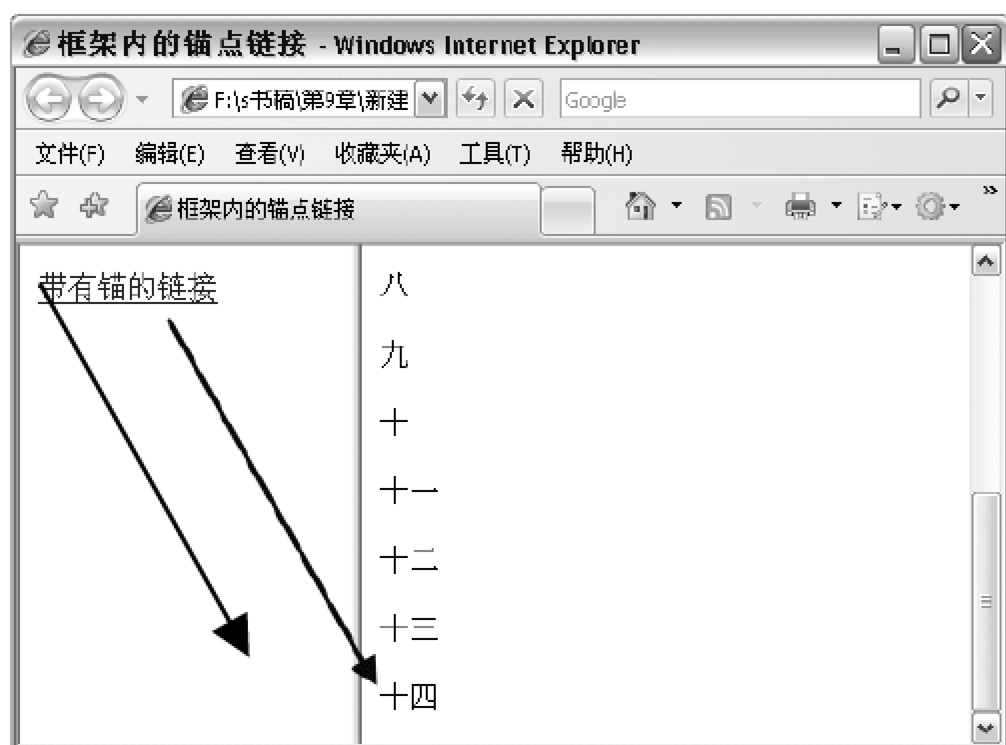


图 9.14　框架内的锚点链接最终页面

【深入学习】重点是代码中的第 8 行，使用#放在链接页面名的后面。这样，当浏览器打开页面时会定位到这一位置，同时，当单击左侧文本“带有锚的链接”时，也会定位到这一位置。

9.5　灵活的<iframe>框架

<iframe>标签的用法不同于<frame>标签，在表现上也有不少差别，<iframe>标签创建的框架具有更好的灵活性。它可以更容易地将框架放在浏览器中的任何位置，可以自由控制窗口的大小，所以，这种框架又被称之为嵌入式框架，或者浮动框架。这种框架也是页面中常见的一种。

<iframe>标签是可以放在<body>标签中使用的，同样，它可以使用修饰框架的属性，常见的样式属性如下：

```
<iframe src="URL"   width="..." height="..."
  align=... marginwidth="..." marginheight="..."
  scrolling="auto" frameborder="...">
</iframe>
```

width 和 height 表明插入框架的宽和高，align 标签表明框架中的内容对齐方式，其他的标签意义和<frame>标签关联的属性是一样的。具体如何使用<iframe>标签，如程序 9.14 创建浮动框架。

【本节示例参考：资料光盘\第 9 章\9-14　创建浮动框架.html】

【实例 9-14】创建浮动框架的方法，其源码展示如下：

程序 9.14　创建浮动框架.html

```
01    <html>
02      <head>
03          <title>创建浮动框架</title>
```

```
04      </head>
05    <body>
06      <a href="http://www.baidu.com.cn" target="three">百度</a>
07      <a href="http://www.google.cn" target="three">谷歌</a>
08      <a href="http://www.soso.com" target="three">搜搜</a>
09    <p>
10      <iframe width="800" height="400"
11       frameborder=0 scrolling="auto"
12       align="center" name="three">      //设置浮动框架的样式属性
13      </body>
14    </html>
```

【运行程序】浏览该页面，本例中使用 target 和 name 属性配合，展示在浮动框架中实现的链接，结果如图 9.15 所示。



图 9.15　创建浮动框架

当单击页面左上角的导航小标题时，浮动框架便链接到所对应的不同页面。第 10 行代码定义了浮动框架的大小，第 11 行表示滚动条自动匹配，框架的边框不显示。第 12 行表明这个框架居于页面中间，并命名为 three。

9.6　案例：制定自己的链接主页

本节将展示如何结合<table>标签来布局页面，思路是通过表格来布局，在表格的单元格中放入浮动框架。同只使用框架集的布局来说，这样还是比较好的，虽然表格布局已经有些落伍了，但是类似这样布局的运用，依然是个不错的方法。如程序 9.15 的表格配合框架的使用。

【本节示例参考：资料光盘\第 9 章\9-15　制定自己的链接主页.html】

【实例 9-15】制定自己的链接主页，其源码展示如下：

程序 9.15　制定自己的链接主页.html

```
01   <html>
02     <head>
03       <title>制定自己的链接主页</title>
04        <style>
05          body {color:black;                          //文本颜色
06                font:2em, arial;                      //文本的字体
07                background-color:silver;              //页面的背景颜色
08                }
09              .biaoti   {font:1.5em,华文琥珀;          //biaoti 对象的字体
10                         color:blue;                  //biaoti 对象的文本颜色
11                         }
12          table td {font:1.2em,;                      //表格的字体
13                    line-height:2.5em;                //表格文本的行距
14                        text-align:center;            //表格中文本的对齐方式
15                        align:center;}                //表格的对齐方式
16              tfoot td{
17                    border-width:0px;                 //页脚的样式
18                    text-align:right;
19                    font-size:12px;
20                    color:green;}
21              a:link    {color : blue;                //链接状态的修改
22                          text-decoration : none;
23                        }
24              a:visited {color : green;
25                          text-decoration : none;
26                          }
27              a:hover {color : red;
28                          text-decoration : underline;
29                          }
30              a:active {color : #999999;
31                          text-decoration : none;
32                          }
33         </style>
34     </head>
35     <body>
36       <table border="0">                   //制定表格
37           <tfoot>
38           <tr>
39                   <td colspan="2">zhaohui </td>
40           </tr>
41           </tfoot>
42           <tr id="biaoti">                  //这两个单元格中放入 logo 图像
43                <td width="100" height="70"><img src="图片/Glass.png" border=0/></td>
```

```
44                  <td class="biaoti">浏览属于自己的主页</td>
45              </tr>
46              <tr>   <!--这两个单元格中放入列表和框架集
47                  <td>
48                      <p><a href=http://www.baidu.com target=three>baidu</a>
49                      <p><a href="http://www.google.cn" target="three">google</a>
50                      <p><a href="http://www.soso.com" target="three">sousou</a>
51                      <p>
52                  </td>
53                  <td>
54                   <iframe width="800" height="430" marginwidth="0"
55                        frameborder=0 scrolling="auto"
56                    align="center" name="three"
57                        src="http://www.baidu.com"
58                      >                       //制定浮动框架
59                      </iframe>
60                  </td>
61              </tr>
62          </table>
63      </body>
64  </html>
```

【运行程序】浏览该页面，最终在浏览器中的显示效果如图 9.16 所示。



图 9.16　制定自己的链接主页

【深入学习】本例使用“田”字型表格布局整个页面。在右下角的单元格中嵌入一个浮动框架，用来显示页面。这是使用表格和框架的组合来布局页面，左侧是导航栏，右侧是页面的主要内容。此外，还可以在左上角放入自己的 logo，右上角放入自己喜欢的页面标题。

通过这个例子，可以设定属于自己的主页。这个方法不难，使用这样的方法，读者可以自行加入更多的页面链接。

9.7 小　　结

本章介绍了通过框架集来布局页面的知识，通过对本章的学习，读者可以将多个页面放入在同一个浏览器窗口中，使页面内容更加丰富，对于浏览者来说，增加了页面的便捷性，无疑也提高了页面的友好度。本章主要的知识点有：

- 使用<frameset>和<frame>标签创建框架。
- 常见的几种布局页面的基本形式。
- 修饰框架的边框和边距。
- 在框架中实现页面链接以及锚点链接。
- <iframe>标签创建浮动框架。

最后通过一个综合案例结合了较多知识点，包括使用样式表来修饰页面内容，结合表格和浮动框架来布局页面。在第 10 章中，将介绍 Web 设计中又一个布局页面的重要元素——层。

第 10 章　当 CSS 样式表遇到层

从本章开始，我们有必要给 HTML 的概念升级。W3C 曾规定："动态 HTML 是一个被某些厂商用来描述可使文档动态性更强的 HTML、样式表以及脚本的结合物的术语。"动态 HTML 指的便是 DHTML（Dynamic HTML）。对大多数人来说，DHTML 便是 HTML 4.0、CSS 样式表以及 JavaScript 的结合物。本章重点介绍的是掌握使用 CSS 样式表的精髓，主要的知识点如下。

- 了解页面是如何布局的。
- 理解层的意义，以及层的特性和它的使用方式。
- 学习创建框模型及它的使用规律和特点。
- 了解 CSS Hack。
- 简单的布局页面的创建。



说明：HTML 4.0 和 CSS 样式表已经在前面章节中详细学习了，JavaScript 的内容将在第 14 章中详细介绍。

10.1　理解块级的意义

在 Web 设计中，设计者使用 CSS 来修饰页面的内容，而与 CSS 这个名字相关的还有一个名字是 CSS-P（Cascading Style Sheets Positioning）。它是 CSS 的一个扩展，表示如何布局页面内容在浏览窗口中的位置。那么使用样式表是如何工作的呢？如程序 10.1 所示为通过层来布局页面。

【本节示例参考：资料光盘\第 10 章\10-1　了解通过层来布局页面内容的位置.html】

【实例 10-1】了解通过层来布局页面内容的位置，其源码展示如下：

程序 10.1　了解通过层来布局页面内容的位置.html

```
01   <html>
02     <head>
03       <title>了解通过层来定位页面内容的位置</title>
04       <style>
05       #navigation {
06       position:absolute;              //定位层在页面中的位置
07       font:1em  幼圆  bold;
08       left:10;                        //层距离窗口 10px 的位置
09       line-height:2em;                //设置行高
```

```
10          width:8em;
11          background:orange;
12          }
13       #content { margin-left:8em;       //设置左边距
14                   font:1.2em arial;
15                   border-left:0.1em orange dotted;
16                   }
17     </style>
18     </head>
19     <body>
20       <div id=navigation>
21     <ul>
22         <li>目录一
23         <li>目录二
24         <li>目录三
25         <li>目录四
26         <li>目录五
27         <li>目录六
28       </div >
29           <div id=content>     绰号其实也讲历史含蕴，短小精悍，很
30        难翻译给其他国家的人听。... ...
31         </div>
32   </body>
33</html>
```

导航栏内容是放在层 navigation 中的

【运行程序】浏览该页面，结果如图 10.1 所示。

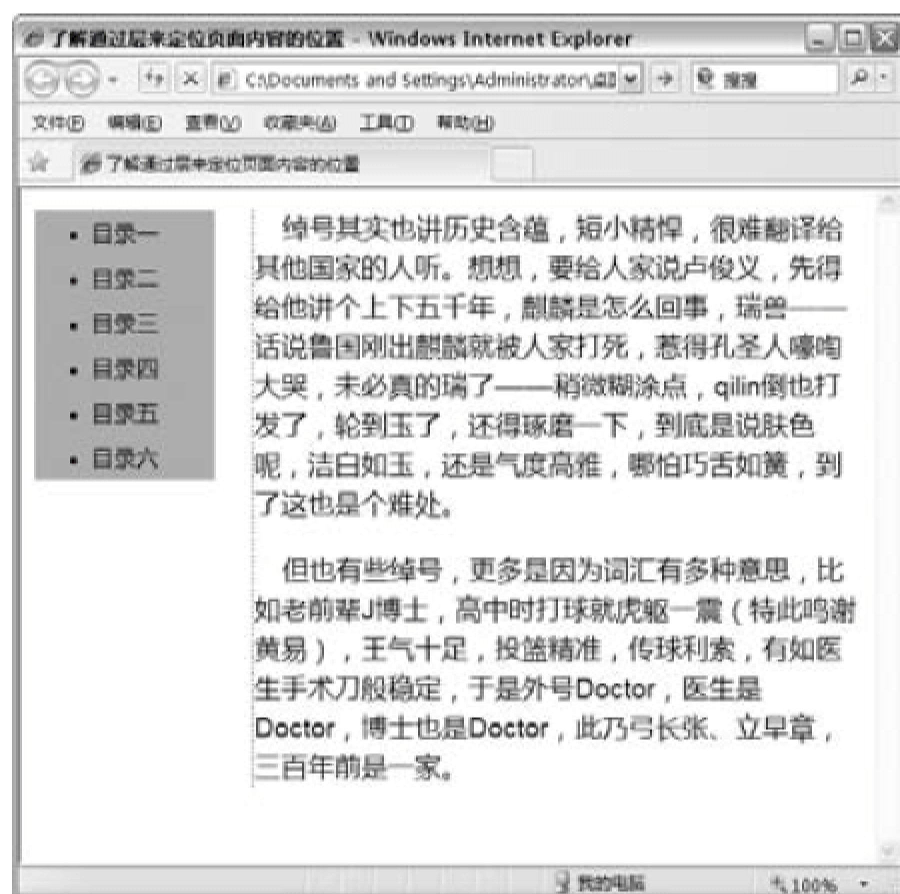


图 10.1　了解通过层来定位页面内容的位置

【深入学习】在本例中，注意第 20 行、28 行、29 行和第 31 行中的<div>标签，可以看成整个窗口是由这两个层布局组成的。两组<div>标签封装了各自对象的样式表，再通过样式表将各自的对象定位在网站中不同的位置。这种布局的思路就是所说的“块级”布局，也是常说的 CSS+DIV。



说明：实例 10-1 中一些对层所设置的样式表，其中的属性作用会在后面的章节中陆续介绍，这里读者只需明白层是如何表现页面的就可以了。

10.2　页面中的层

CSS 样式表可以通过被封在层上这种方式来限制页面所修饰的内容。这就是说，为了使用样式表只是对页面局部起修饰作用，于是设计者将页面的局部内容定义在某个范围中，这个范围便称之为 CSS-layer（CSS 层）。CSS 层可以通过 HTML 标签来定义，这种使用方法是 Web 设计中的一枚利器。

10.2.1　行<span>和层<div>

设计者常把 CSS 样式表放在<span>和<div>这两个布局页面的标签中，其作用是令样式表可以只对标签中的内容元素起作用。这两者的区别是<span>标签针对的是行内的对象，而<div>标签针对的是层内的对象。为了形象说明什么是行标签，这里通过以下代码展示一个简单的案例：

```
<style>
#color {color:red;
        }
</style>
《烬余录》像是一个历尽沧桑的百岁老人所写，<span id="color" >但是当时的张爱玲只有 24 岁。</span >她对自己的自私和冷酷，有一种抽离。
```



说明：为了节约书面空间，这部分代码只是给出了关键部分。这里定义了一个 color 的样式表，通过 id 选择器作用于<span>标签内的文本。

结果如图 10.2 所示。

《烬余录》像是一个历尽沧桑的百岁老人所写，但是当时的张爱玲只有24岁。她对自己的自私和冷酷，有一种抽离。

图 10.2　<span>标签的作用

通过<span>标签的作用，这行文本只有<span>标签中间的内容被修饰了。<div>标签如程序 10.1 中所示。如代码第 20～28 行中的内容，仅修饰了页面中的左侧导航栏，这就是<div>标签的作用。



注意：如果<div>标签中没有引用样式表，那么，其作用就相当于<p>标签。

10.2.2　层的基本定位

通过一些基本的属性可以给层定位在页面中的任何位置，这些主要的属性有方位属性，如层的左、右、上、下。描述大小的属性，如层的宽、高、参照位置。

- left：相当于窗口左边的位置。
- right：相当于窗口右边的位置。
- top：相当于窗口上边的位置。
- bottom：相当于窗口下边的位置。
- width：表示层的宽度。
- height：表示层的高度。
- position：用来控制采用什么样的方式定位图层。

position 属性下可以定义为 absolute、relative 和 static 属性。absolute 属性表示为“层的位置以网页的左上角为基准来设置”；relative 表示为“层的位置以其原始值的位置来设置”；static 表示为“层的位置以 HTML 默认的位置来设置”。

事实上，只要通过 left 和 top 属性就可以控制层在页面中的位置了，而 width 和 height 属性设置层的大小，如程序 10.2 所示。

【本节示例参考：资料光盘\第 10 章\10-2　层的基本定位.html】

【实例 10-2】层的基本定位，其源码展示如下：

程序 10.2　层的基本定位.html

```
01  <html>
02      <head>
03         <title>层的定位</title>
04      </head>
05      <style>
06        div {position:absolute;   //以浏览器窗口为基准设置
07              width:300px;
08              height:300px;
09              left:5em;            //距离窗口 5em（em 参考 7.3.2 小节）来定位页面内容的位置
10              top:5em;
11              }
12      </style>
13      <body>
14      <div>
15  夕阳武士为了筹措盘缠回故乡而出战马贼，但小时医生的告诫他会在这一年失明，虽然光线在
16  他的眼中已经日渐昏暗，但他还是坚持出战最终战死。欧阳锋很奇怪他为何要急着赶回去？武
17  士回答说家乡的桃花很美，他要回去看桃花！欧阳锋好奇去了武士的家乡，发现那里根本没有
18  桃花，只有一个女人，她的名字叫做桃花......
19     </div>
20     </body>
21  </html>
```

【运行程序】浏览该页面，结果如图 10.3 所示。



图 10.3　层的定位

【深入学习】如图 10.3 中所示，上面距离窗口顶部是 5em，框架距离窗口左侧是 5em，边框的长度和宽度都是 300px。如果需要把很多个层放在页面中，这就是一个最简单的布局页面的方法。



注意：在层中文本内容不足 height 属性下的高度时，层的大小会默认为 300px。如果层中的内容超出 height 属性下的高度时，浏览器会自动修改原始层的高度来适应文本的内容。

10.2.3　层的叠加

层不同于表格和框架，其最大优势在于可以叠加。这是因为层具有一个“Z 轴”的特性，Z 轴好比 3D 坐标中的 Z 轴，是一个上下层级的关系，就是说一个层是可以覆盖在另一个层上面，如图 10.4 所示。

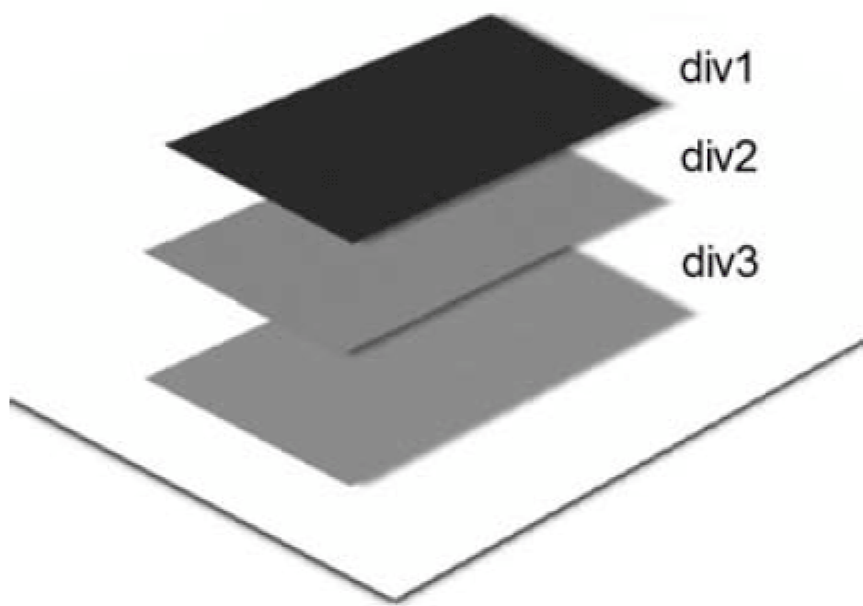


图 10.4　层叠示意图

如果有 3 个层如图 10-4 中一次堆叠，浏览器中最终显示的是 div1 层中定义的内容。它的叠加规律是这样，在<body>标签中，根据层依次出现的顺序，以此来判定层的上下级关系。定义具体的使用如程序 10.3 所示。

【本节示例参考：资料光盘\第 10 章\10-3　层的叠加.html】

【实例 10-3】层的叠加，其源码展示如下：

程序 10.3　层的叠加.html

```
01   <html>
02     <head>
03       <title>层的叠加</title>
04     </head>
05     <style>
06      div {height:300;
07           width:300;
08           }
09      #d1 {position:absolute;                    //定位页面层位置属性
10           background-color:green;
11           left:2em;                             //距离窗口 2em 来定位页面内容的位置
12           top:2em;
13           }
14       #d2 {position:absolute;
15            background-color:blue;
16            left:4em;                            //距离窗口 4em 来定位页面内容的位置
17            top:4em;
18            }
19        #d3 {position:absolute;
20             background-color:red;
21             left:6em;                           //距离窗口 6em 来定位页面内容的位置
22             top:6em;
23             }
24     </style>
25     <body>
26       <div id="d1">                           //定义这个为最下面的层
27       <div id="d2">                           //定义这个为中间的层
28       <div id="d3">                           //定义这个为最上面的层
29     </body>
30   </html>
```

【运行程序】浏览该页面，结果如图 10.5 所示。



图 10.5　层叠的关系

【深入学习】在本例中，首先写入的是代码第 26 行，表明最先放入页面的是 d1 层，代码第 28 行最后写入，即表明最后放入页面的是 d3 层，所以，层的叠加顺序是，d1 层在最下面，而 d3 层在最上面。

如果设计者希望在页面中改变叠加的顺序而不受先后定义的约束，可以使用 z-index 属性，这个属性表明层在 Z 轴上的叠放位置。上述代码中，如果在任何一块样式表中添加 z-index:1，如代码 19 行中 d3 样式表中添加的，那么 d3 块的层将会被排放在最下面一层。

10.3 框 模 型

层的内部便是一个框模型（box model），这个概念很重要。在 CSS 广泛应用之前，建立一个出色的页面布局只能通过框架集、表格，大量内嵌表格框架，或者一堆堆的<p>标签和空格符号。而当使用层的框模型思路布局时，设计者们就找到了最好的选择。有时它完全可以替代框架、表格等。这种方法不仅可以为页面代码精简，而且大大缩短了页面的刷新时间，这样更易于管理代码。

10.3.1 理解框模型

层中内容的外面被很多空间级概念的物质包围，如空距（padding）、边框（border）和边距（margin）。页面中任意一个层中内容的周围理论上是这样被包围的，如图 10.6 所示的框模型。



图 10.6 框模型

页面内容可以是任何内容。如果设计者愿意，可以用任何东西替代页面内容，一段文本，或者一幅图像、一个表格，甚至是框架集。当然也可以是一个层。而为了精确布局页面，了解框模型的大小在页面中的表现，可以参考程序 10.4 给出的一个框模型的简单实例。

【本节示例参考：资料光盘\第 10 章\10-4 框模型的大小.html】

【实例 10-4】框模型的大小，其源码展示如下：

程序 10.4 框模型的大小.html

```
01  <html>
02    <head>
03      <title>框模型的大小</title>
04    <style>
```

```
05  #a {background-color:teal;
06          width:20em;
07          height:10em;
08          padding:1em ;                    //空距的距离是 1em
09          border:1em solid navy;           //边框的宽度是 1em
10          margin:1em;                      //边距的宽度是 1em
11            }
12  #b {background-color:teal;
13        width:20em;
14        height:13em;
15        padding:3em;                       //空距的距离是 3em
16        border:1em solid navy;             //边框的宽度是 1em
17        margin:3em;                        //边距的宽度是 3em
18            }
19        </style>
20      </head>
21      <body>
22          <div id="a">
23  《烬余录》像是一个历尽沧桑的百岁老人所写，但是当时的张爱玲只有 24 岁。她对自己的自私
24  和冷酷，有一种抽离。
25        </div>
26          <p>
27        <div id="b">
28  《烬余录》像是一个历尽沧桑的百岁老人所写，但是当时的张爱玲只有 24 岁。她对自己的自私
29  和冷酷，有一种抽离。
30        </div>
31      </body>
32  </html>
```

【运行程序】浏览该页面，结果如图 10.7 所示。

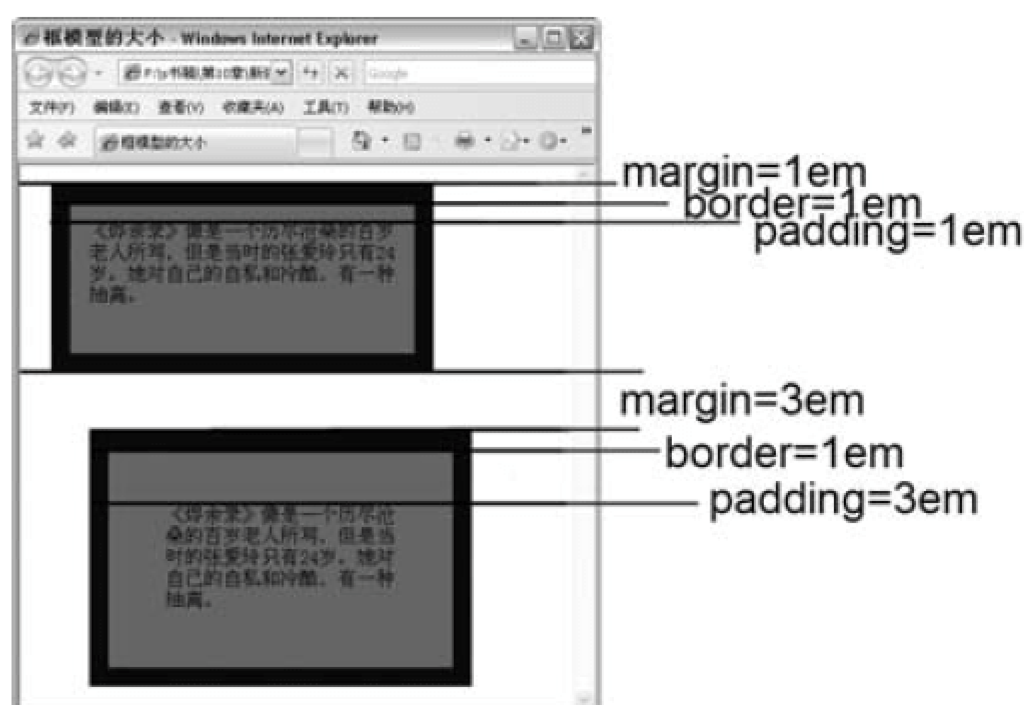


图 10.7　框模型的大小



注意：在这里 height 属性值是指从边框上沿（border）到边框下沿的距离。

【深入学习】如图 10.7 中所示，层实际的高度是“层中页面内容的高度+上空距+上边框+上边距+下空距+下边框+下边距”的值。

第一个层中，如果事先设定的 height 属性值超出层中上下边框之间的距离，那么页面内容会以空白部分填补缺少的高度，最终令层的上下边框之间的距离等于 height 属性值。如图 10.7 中上面第一个层，页面内容即文本的高度为 4em，上边框+上空距+下边框+下空距为 4em。这样实际层的上下边框之间的距离是 8em，而 height 属性设置为 10em，所以层中文本内容会补齐 2em 高度。

而第二个层设置的高度正好符合上下边框的距离。所以层中文本没有出现多余的空行。如果 height 的属性值小于层的上下边框之间的距离，这时 height 属性值便失去作用。例如，样式表“#b”中，height 设置为 10em，这样的话，height 值将不起作用。第二层的样式不会发生改变。

此外，如果上下两个层之间放在一起，那么，两个层的边距（margin）将会出现合并，最终以较大的边距为标准。如图 10.7 所示，上下层之间的距离是 3em，而并非是 3em+1em 得到 4em。



注意：页面内容占据的空间是由 width 和 height 属性设置的，而内容周围的边距 padding 和边框 border 值是另外计算的。但是在 IE 浏览器中，width 和 height 属性是包括边距和边框的长度的。

10.3.2 空距 padding 属性

padding 属性又常被称为内边距。padding 属性可以细分为 padding-top、padding-right、padding-bottom 和 padding-left 4 个属性，通过它们可以控制一个框模型中的每一边空距，如“padding-bottom:1.5em;”。此外为了方便，设计者可以使用快捷的写法来分别设置 4 条边，如下所示。

- 当 padding 只定义一个数值时，例如：

```
padding:1em;
```

表示所有框模型空距为 1em。

- 当 padding 定义两个数值时，例如：

```
padding:1em 2em;
```

表示所有框模型空距顶边和底边为 1em，左边和右边为 2em。

- 当 padding 定义 3 个数值时，例如：

```
padding:1em 2em 3em;
```

表示所有框模型空距右边为 1em，左边和右边为 2em，底边为 3em。

- 当 padding 定义 4 个数值时，例如：

```
padding:1em 2em 3em 4em;
```

表示所有框模型空距将按照顺时针方向，由顶边为 1em 开始，依次是右边为 2em、底边为 3em 和左边为 4em。

此外还有一个有趣的用法，就是借助 padding 属性可以使用自定义图像来作为空距。但是在浏览器中这种方法只能定义其中的一条边，如程序 10.5 中使用自定义图像来修改边距的样式。

【本节示例参考：资料光盘\第 10 章\10-5　使用自定义图像来作为空距.html】

【实例 10-5】使用自定义图像作为空距，其源码展示如下：

程序 10.5　使用自定义图像来作为空距.html

```
01  <html>
02    <head>
03      <title>使用自定义图像来作为空距</title>
04    <style>
05      body {text-align:center;
06              line-height:20px;                         //设置文本的行高
07              white-space: pre;                         //相当于 pre 标签的作用
08              }
09      h1 {font:1.5em 宋体;
10          }
11      h2{font:1em 幼圆;
12          }
13      #c {margin-left:40px;                             //定义层中文本的位置
14          padding-left:20px;                            //定义左边空距的位置
15          background:url(图片/回形针.png) left repeat-y;   //使用自定义图像设置左边空距
16          }
17      </style>
18    </head>
19    <body>
20      <div id="c">
21        <h1>将进酒</h1>
22        <h2>李白</h2>
23  <br>君不见黄河之水天上来，奔流到海不复回。
24  <br>君不见高堂明镜悲白发，朝如青丝暮成雪。
25  <br>人生得意须尽欢，莫使金樽空对月。
26  <br>天生我材必有用，千金散尽还复来。
27  ......
28      </div>
29    </body>
30  </html>
```

【运行程序】浏览该页面，结果如图 10.8 所示。



说明：代码第 7 行中“white-space: pre;”的作用是保持 HTML 源代码的空格与换行，等同于 pre 标签。在 IE 浏览器中不能得到支持，所以使用了 Firefox 浏览器，这里为了精简代码使用了这一标签，在大量代码的页面中不推荐使用。

【深入学习】图 10.8 中左侧一栏便是使用自定义图像设置的。由于这种使用方法只能定义 4 条边中的一条，所以在框模型中就显得不那么实用。如果希望制作 4 条边自定义图像的边框，更好的办法

是使用已经做好的 PNG 图像或 GIF 图像。

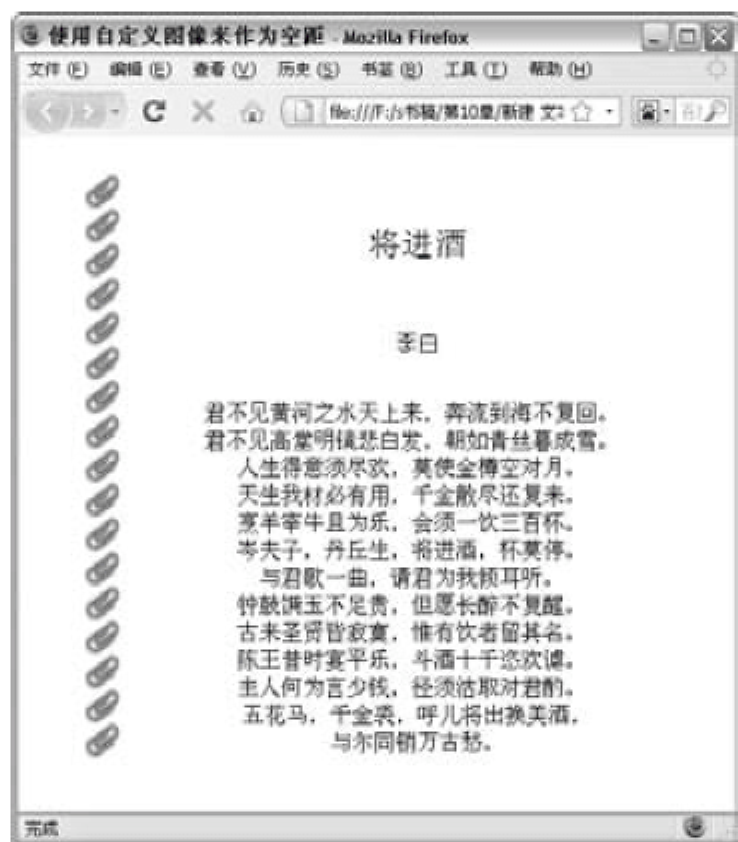


图 10.8　使用自定义图像来作为空距



说明：padding-bottom 可以用来修饰超链接下划线，具体可参考第 6 章的内容。

10.3.3　边框 border 的扩展属性

border 是一种使用频率非常高的属性，表格、边框中都有它的身影。对于边框，不仅可以改变它的宽度，而且可以指定其格式和颜色（参考第 8 章的内容）。所以，边框的属性具有多样式的扩展，其属性可以细分为 border-width 属性、border-style 属性和 border-color 属性。

- border-width：表示边框的宽度。
- border-style：表示边框的样式，常用的有 solid、dotted、dashed 等。
- border-color：表示边框的颜色。



注意：默认情况下，边框的属性是 none。就是说，如果像 border:2px 代码这样，那么是不会有边框的，如果前面已经有设定过的边框样式，这句将默认继承上级的边框样式。

同样，也可以像 padding 属性那样，采用快捷的方式来定义边框，例如：

```
border:3px dotted red;
```

或者使用快捷方式定义每一条边框，例如：

```
border:1em 2em 3em 4em;
```



说明：CSS 2.1 指出，页面内容的背景是由内容、内边距和边框区的背景组成的。大多数浏览器都遵循 CSS 2.1 定义，不过一些较老的浏览器可能会有不同的表现。

10.3.4　边距

margin 属性又称之为外边距，就好像是围绕在边框范围外的一层“气场”。padding 属性值不能为负值，而 margin 属性值可以为负数，以此对内容进行叠加。前面已经有所提及，如果两个或两个以上的纵边边距紧挨在一起，那么彼此相邻的边距会发生合并现象，最终彼此边框之间的距离为两个边距较大的边距宽度，而非两个边距之和。

类似于空距和边框，边距属性也可以细分为上、下、左、右 4 条边分别控制。它们是上边框属性 margin-top、下边框属性 margin-bottom、左边框属性 margin-left 和右边框属性 margin-right。

此外，如果两个不同页面内容的边距彼此上下相邻，也会发生合并的现象。为了解决这个问题，可以在其中的一个页面内容中添加边框或者空距，来隔开两个边距接触，如程序 10.6 所示。

【本节示例参考：资料光盘\第 10 章\10-6　不同页面内容的边距相邻.html】

【实例 10-6】不同页面内容的边距相邻，其源码展示如下：

程序 10.6　不同页面内容的边距相邻.html

```
01   <html>
02     <head>
03       <title>不同页面内容的边距相邻</title>
04      <style>
05       #d {margin:2em;                              //边距设置为 2em
06           background:cyan;
07           border:1px solid;
08           }                                        //设置边框的样式
09          p { margin:1em ;                          //边距设置为 1em
10             background:yellow;
11             }
12            </style>
13            </head>
14            <body>
15              <div id="d">
16                <p>
17        曾听过一个故事…
18             </div>
19           </body>
20         </html>
```

【运行程序】浏览该页面，结果如图 10.9 所示。

【深入学习】如果这段代码中缺少了第 7 行的代码对边框的设定，层（div）的边距和文本的边距彼此相接，则层的边距会被合并，如图 10.10 所示。

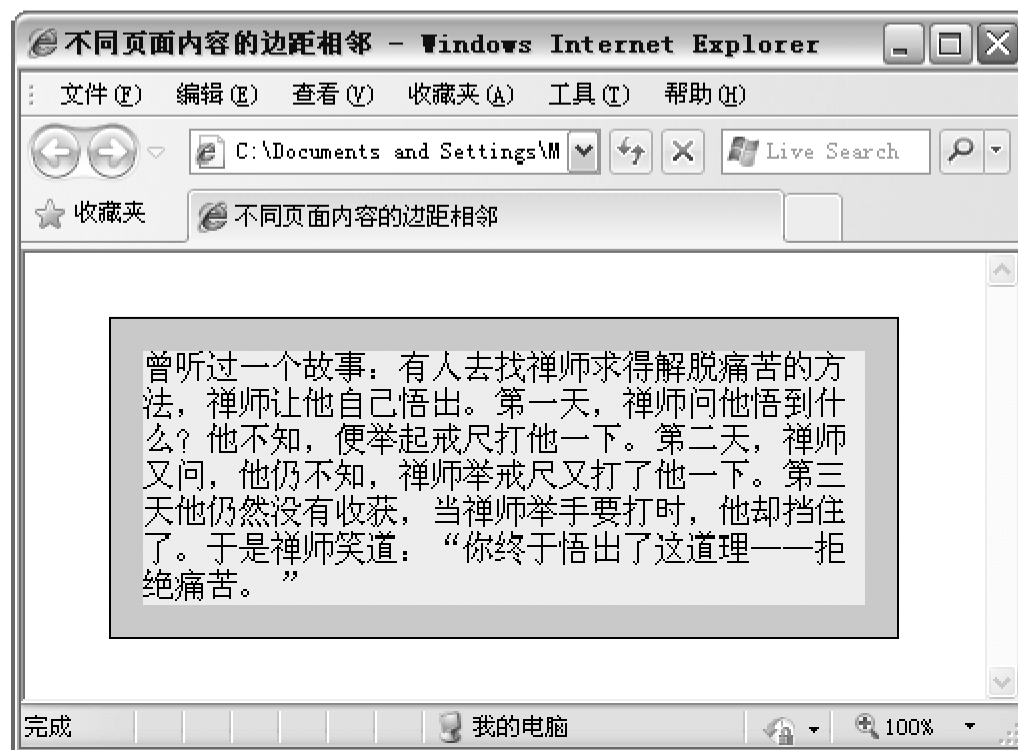


图 10.9　不同页面内容的边距相邻

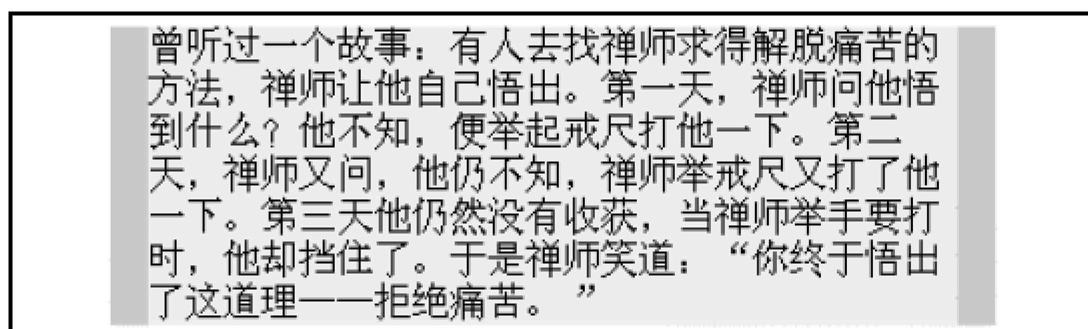


图 10.10　错误的合并方式

10.3.5　框模型的溢出

如果层中的内容太多，以至于超出了层的初始设定范围，IE 浏览器此时会“自作主张”地帮忙拉伸层的范围。为了改变这种情况，令层的大小不会发生改变，可以使用 overflow 属性。在默认情况下，overflow 属性为 visible，意思是页面内容都是可见的。所以，这是层的大小失去控制的原因，如程序 10.7 的 overflow 属性所示。

【本节示例参考：资料光盘\第 10 章\10-7　使用 overflow 属性.html】

【实例 10-7】使用 overflow 属性，其源码展示如下：

程序 10.7　使用overflow属性.html

```
01   <head>
02     <title>使用 overflow 属性</title>
03       <style>
04       #d {margin:2em;                    //设置边距大小
05           height:20em;                   //设置页面高度
06           width:30em;                    //设置页面宽度
07           background:cyan;               //设置页面背景颜色
08           overflow:auto;                 //自动控制页面的滑动条
09             }
10         </style>
11     </head>
12     <body>
13      <div id="d">
```

```
14          <p>
15          对冲基金（Hedge Fund）
16            ......
17        </div>
18      </body>
19   </html>
```

【运行程序】结果如图 10.11 所示。它像浮动框架一样内嵌在页面中。从这点来说，CSS 2.0 之后所进行的页面设计，使用对层的框模型创建使框架集的使用大大降低。

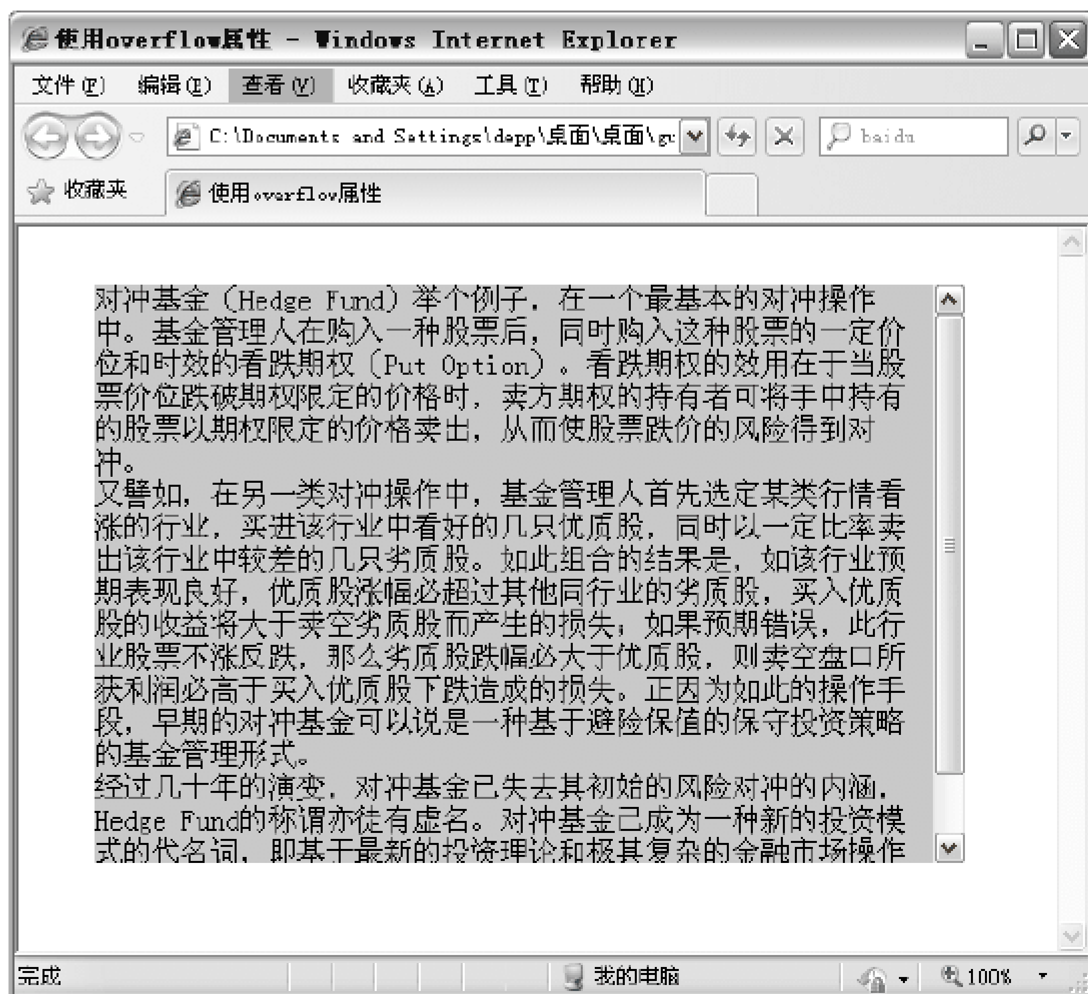


图 10.11　使用 overflow 属性

【深入学习】第 8 行代码将 overflow 属性设置为 auto，则窗口会根据页面内容的多少来决定是否出现滚动条。此外，还可以设置为 hidden 和 scroll 属性，前者会严格按照属性性质来设定框的大小，超出的内容将会被隐藏。而后者页面内容无论是否溢出框架，都将显示滚动条。

10.4　定制层的 display 属性

前面已经了解到，层的表现是通过“框”这种结构，框可以是块级对象（block element），也可以是行内对象（inline element）。那么所谓 display 属性就是用来控制其中的内容是块级还是行级。所以，基本的定义为 block，表现为块级；或者定义为 inline，表现为行级。默认情况下是 none，表现为不显示框，代码如下：

```
display:block;
```

在第 7 章的内容中已经讲解了使用 display 属性来设计导航栏的方法。其中的原理正是因为 display 属性的行内 inline 的作用，设计者可以将框中任何一个对象设置成所选择的类型，如程序 10.8 展示的 display 属性所示。

【本节示例参考：资料光盘\第 10 章\10-8　display 属性.html】

【实例 10-8】display 属性，其源码展示如下：

程序 10.8　display属性.html

```
01  <html >
02  <head>
03       <title>display 属性</title>
04       <style type="text/css">
05            body  {
06                   text-align:"center";
07                   font: 80%  黑体;
08                   }
09            h1 {font-size: 2em;                       //设置字体大小
10                 }
11            h2 {font-size: 1.5em;
12                 }
13            #kuai, #hang {background: silver;
14                          border: 2px solid black;    //#kuai 和#hang 对象的边框样式
15                          }
16            span {   background: white;
17                     display: block;                  //设置为块级对象
18                     border: 0.5em dashed green;      //设置 span 对象边框的样式
19                     padding: 1em;
20                     margin: 0.5em;
21                     }
22            span.yanse {
23                     background: yellow;              //行内对象的背景颜色
24                          }
25            #hang span {
26                     display: inline;                 //设置为行内对象
27                          }
28       </style>
29    </head>
30    <body>
31       <h1 >“块”  和  “行”  </h1>
32       <h2>块</h2>
33         <p id="kuai"><span>北京</span><span class="yanse">上海</span><span>香港
34         </span><span class="yanse">海南</span></p>
35       <h2>行</h2>
36         <p id="hang"><span>北京</span><span class="yanse">上海</span><span>香港
37         </span><span class="yanse">海南</span></p>
38    </body>
39  </html>
```

【运行程序】浏览该页面，结果如图 10.12 所示。

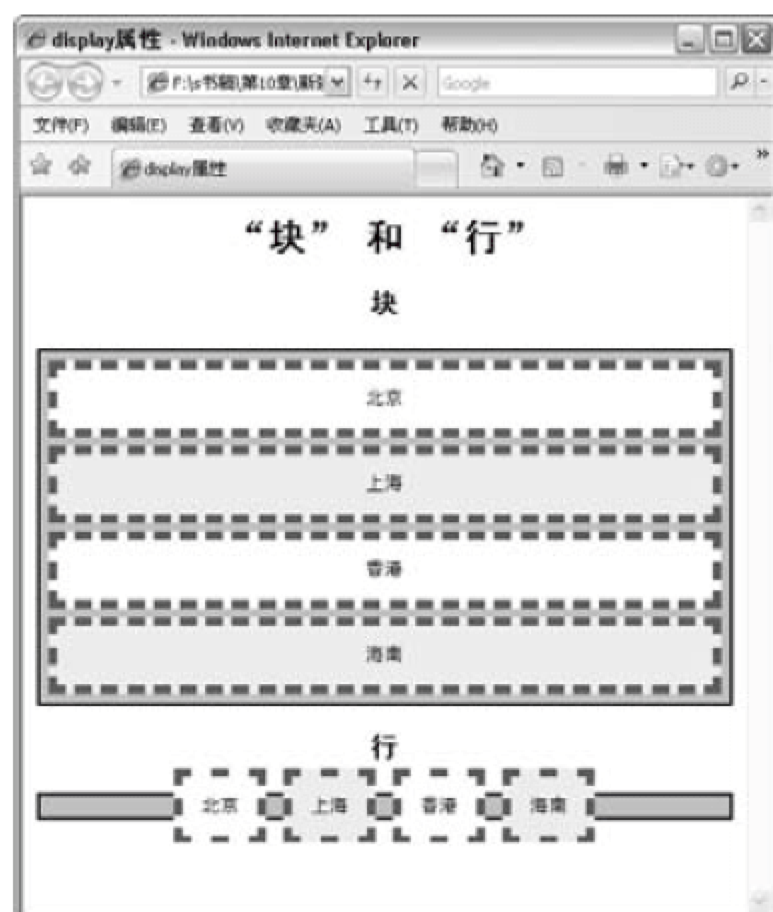


图 10.12　display 属性



注意：定义为行内对象的框模型，在默认情况下将忽视纵方向上的边距。如图 10.12 所示，空距和边框将会超出所在的行。

【深入学习】代码第 33 行和 34 行定义了其中的文本为块级对象，引用了 kuai 这个样式表。在页面中可以看到，<span>标签内的文本是以纵排依次排列，就是块级样式。而代码第 36 行和 37 行中的内容为 hang 样式表的对象。在页面中的结果如图 10.12 所示，陈列出“行”的样式依次横向排列。

为了防止这种溢出的情况，inline 属性扩展出 inline-block。顾名思义，这个属性是为了将行内对象中的内容定义为“行内框”。所以，如果将代码第 26 行改为“display:inline-block;”，那么将表现为如图 10.13 所示的效果。

行



图 10.13　inline-block 属性

10.5　CSS Hack

目前世界上流行着多种浏览器，主要有 IE 浏览器、Firefox 浏览器、Opera 浏览器以及 Google 浏览器，它们基于不同的内核，对 CSS 的解析也就不一样，这直接导致生成的页面效果不同。例如，最直接的影响就体现在框模型中对距离的理解，这对设计者来说，是很伤脑筋的一件事。怎样才能够解决浏览器兼容的问题呢？只能针对不同的浏览器写不同的样式表，这种写法被称之为 CSS Hack。

尽管有许多 Hack 针对不同的浏览器提供了解决方案，如在解决 IE 浏览器和 Firefox 浏览器中布局不同的问题时，常用的一个是!important。由于!important 不被 IE 支持，而其他浏览器可以支持，使用

这个特性可以用来解决很多问题。如有时在定义 padding 对象时，在 IE 浏览器和 Firefox 中有误差，则设计者先制定被非 IE 浏览器能识别的声明，再添加一个 IE 也能识别的声明。其中需要注意 CSS 的先后声明顺序，下面通过程序 10.9 来说明如何使用 Hack。

【本节示例参考：资料光盘\第 10 章\10-9　使用 Hack.html】

【实例 10-9】使用 Hack 的方法，其源码展示如下：

程序 10.9　使用Hack.html

```
01  <html>
02    <head>
03      <title>CSS Hack</title>
04      <style>
05          .select {border:20px  solid navy !important;        //设置 Hack 中的边框样式
06                  width:230px !important;                     //设置 Hack 中的宽度
07                  padding:20px !important;                    //设置 Hack 中的空距
08                  border:20px  solid orange;                  //设置边框样式
09                  width:300px;                                //设置宽度
10                  padding:20px;                               //设置空距
11                  font;1.5em 新宋体;
12                  text-align:center;
13                }
14      </style>
15    </head>
16    <body>
17      <div class="select">在 Firefox 中的效果是蓝色边框，它的 width 设置为 230px。
18  而在 IE 7 浏览器中的效果是橙色边框，它的 width 设置为 310px。</div>
19    </body>
20  </html>
```

【运行程序】浏览该页面，结果如图 10.14 所示。

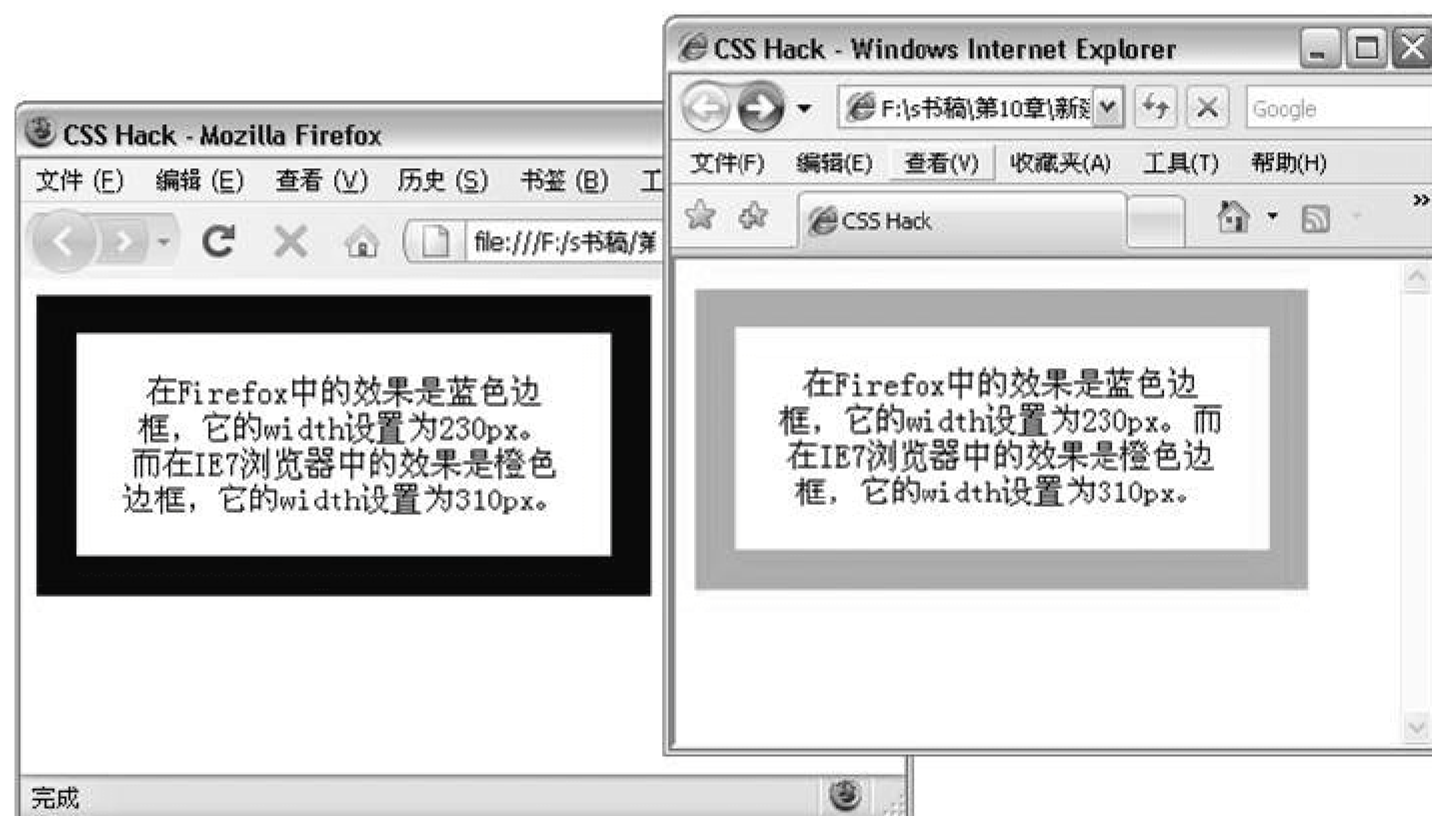


图 10.14　使用 CSS Hack 的页面效果

【深入学习】图 10.14 中左侧是 Firefox 浏览器中的页面效果，右侧是在 IE 7 浏览器中的页面效果。如代码第 5～7 行中，由于声明中附加了!important，所以 IE 浏览器无法分析声明。而 Firefox 接受了信息，所以在浏览器中显示出蓝色的边框。IE 浏览器接受了第 8～10 行中的代码声明，所以在浏览器中显示出橙色边框。

此外，从效果上看，两个边框的大小都是一样的。但是在代码中可以看到，它们被设置了不同的宽度，造成这种不同的原因就是因为浏览器之间存在差异。但对设计者来说，很糟糕的是，他们可选择的方法不多，只能花些时间和耐心，去测试不同的页面效果进行调整。所以设计者在制作页面时，要尽量避免使用 Hack。

对于浏览器的不断升级和创新，CSS Hack 也在不断地变化之中，也许哪天某一标签的作用将被所有浏览器统一，又或许出现新的技术而某些浏览器无法使用。作为设计者，也只能希望 W3C 组织来统一 Web 的标准。

10.6　案例：简单的 CSS+DIV

从本章开始，读者应该习惯当谈到页面布局这样的问题时，首先想到的是使用 CSS 和层的配合使用，这里使用前面所学的知识制作一个简单的三栏式布局的页面，如程序 10.10 所示。相当于本章开始的两栏布局，三栏布局并不只是多加了一个层。在本案例中，要留心三栏的布局在数字上的巧妙安排，如果数字排得不够精确，页面也许会变得面目全非。

【本节示例参考：资料光盘\第 10 章\10-10　三栏布局下的页面.html】

【实例 10-10】三栏布局下的页面，其源码展示如下：

程序 10.10　三栏布局下的页面.html

```
01  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
02          "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
03  <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
04     <head>
05          <title>使用 CSS+DIV 布局页面</title>
06          <style type="text/css" >
07                #header {background: silver;                //页面头部的样式表
08                          }
09                 #footer {background: gray;                 //页面底部的样式表
10                           }
11                 body {font: 80%   微软雅黑;
12                        margin: 0;
13                        }
14                h1,h2 {margin-top: 0;
15                        }
16                #oneBlock {font-size:1.5em ;                //页面主题左侧的导航栏
17                           position: absolute;
```

```
18                          left: 0;                          //距离页面左侧的距离为 0
19                          padding:20px;
20                          width: 130px; height:300px;
21                          background-color:orange;
22                                }
23              #twoBlock {font-size:1.5em;                    //页面主题左侧的导航栏
24                              position: absolute;
25                              right: 0;                      //距离页面右侧的距离为 0
26                              padding-right:10px; padding-top:20px;
27                              width: 135px;height:500px;
28                              background-color:orange;
29                             }
30              #content   {text-indent:25px;                  //首字母缩进
31                             margin: 0 180px 0 200px;
32                             }
33          </style>
34      </head>
35      <body>
36                <div id="header">使用 CSS+DIV 布局页面</div>
37                <div id="oneBlock">
38                    <ul>
39                         <li><a href="">北京</a></li>
40                         <li><a href="">上海</a></li>
41                         <li><a href="">香港</a></li>
42                    </ul>
43                </div>
44                    <div id="twoBlock">
45                           <ul>
46                          <li><a href="">Beijing</a></li>
47                               <li><a href="">Shanghai</a></li>
48                               <li><a href="">Hongkong</a></li>
49                           </ul>
50                    </div>
51                        <div id="content">
52                                <h1>中国</h1>
53                                <h2>北京</h2>
54              北京位于华北平原与东北平原、蒙古高原的交接处，
55   ......
56                                <h2>上海</h2>
57              原来的上海只是一个以渔业和棉纺织手工业为营的小镇，19 世纪
58   ......
59                                <h2>香港</h2>
60              香港是国际的金融中心之一，金融机构和市场紧密联系。
61   ......
62                       </div>
63            <p>
```

```
64                <div id="footer"> 
65                </div>
66        </body>
67    </html>
```

【运行程序】浏览该页面，代码运行结果如图 10.15 所示。

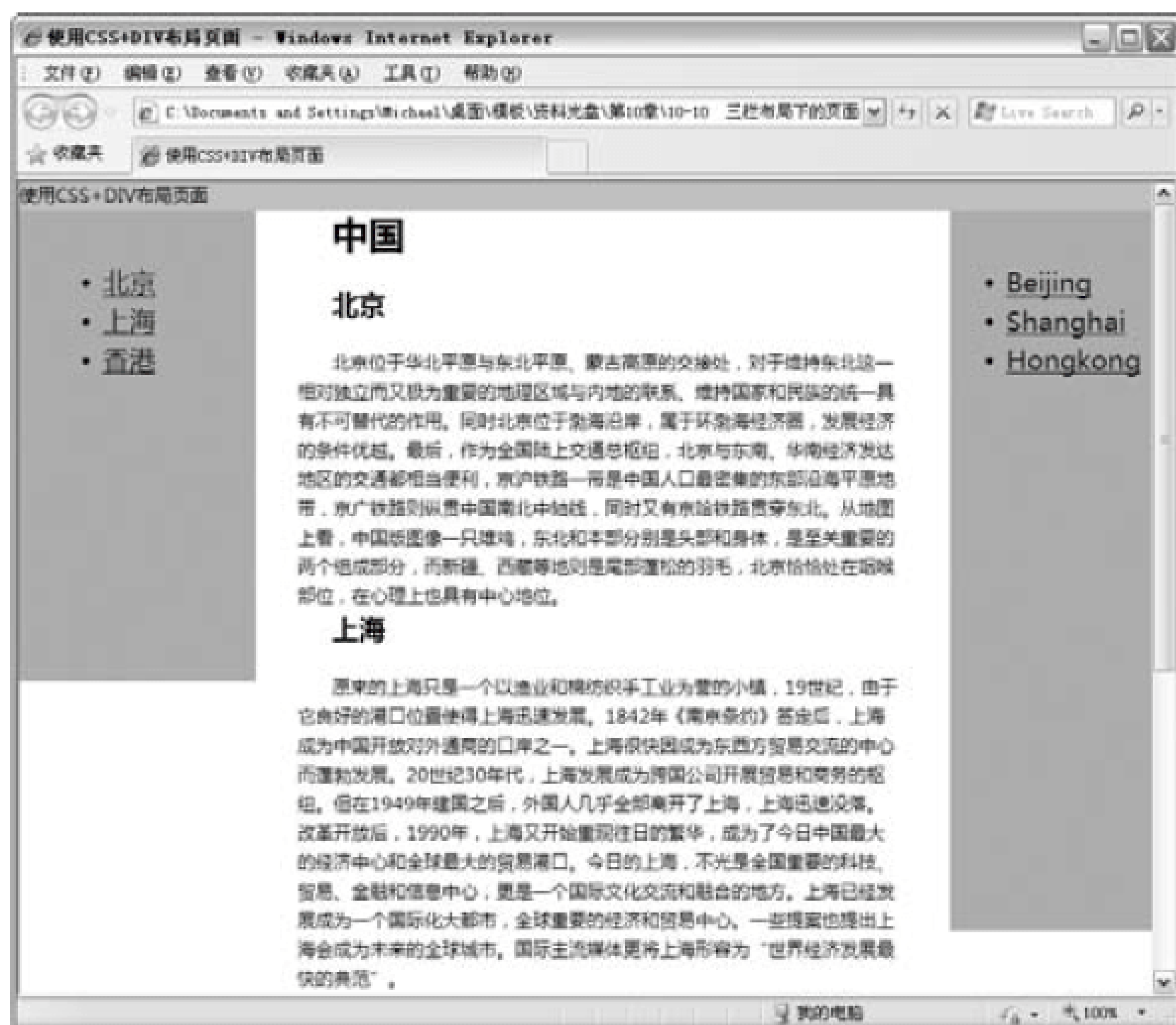


图 10.15　三栏式布局

【深入学习】在本例中，代码第 7～9 行中的两个样式表#header 和#footer 首先定义了页面中头尾的 Banner。该例的难点在于代码第 16～22 行的样式表#oneBlock 和第 23～29 行的#twoBlock 样式表，其重点在于 width 的长度设置。oneBlock 是一个宽 130px、高 300px 且居左为 0 的框模型。twoBlock 是一个宽 135px、高 500px 且居右为 0 的框模型。所以之后#content 定义的 margin 距离页面左右的距离分别为 200px 和 180px。这个数值没有绝对的精确度，只要保证不和左右两个块级区域重合就可以了。

其中 padding 属性用来控制文本在框模型中的位置，默认情况下是以左边为标准。所以在#twoBlock 对象中，不得不使用细化的 padding 属性以便于控制文本的位置。



说明：在样式表中，position 属性用来定位框模型的位置，这是一种绝对定位的用法，具体的使用将在第 11 章中详细介绍。

10.7　小　　结

本章介绍了 CSS+DIV 布局的入门知识及一些基础的概念型知识。这些基础知识很重要，了解了这些知识点后，在今后 Web 设计的学习中，即使是自学，也可以很好地融会贯通新的知识。本章主要的知识点有：

- 使用 CSS 样式表来布局页面，使页面分割成块级结构。
- 理解层的意义，它可以用来封装 CSS 样式表，以及层的特性和使用方式。
- 层的布局格式典型如框模型，了解空距、边框、边距的含义和特性。
- 了解 CSS Hack，使用它来解决浏览器的兼容性。
- 使用<display>属性来设置块级对象和行内对象。

在第 11 章中，将进一步讨论页面布局，以及如何令模块实现在页面中的定位等问题。

第 11 章　进一步讨论页面布局的方法

在了解了基本的层和框模型后，接下来便可以进一步去讨论可以做出怎样效果的页面。设计者需要将页面划分为不同的区域，就好像是在城市规划，将不同的页面内容布局在不同的位置，哪里放置公园，哪里放置社区，哪里是商业区等，页面的分布和内容所处的位置，将会给浏览者传递重要的网站讯息。本章主要的知识点如下。

- ❑ 了解框模型的定位以及使用它们来布局页面。
- ❑ 浮动层的使用。
- ❑ CSS 3.0 的一些奇特技术以及未来发展趋势。
- ❑ YAHOO 的 YUI Grid CSS。

11.1　页面中的定位

当能够娴熟地将层布局在页面中时，Web 设计一定会带给用户无限的乐趣和享受过程带来的惊喜。在第 10 章中，定义框模型时，已经了解了有一个 position 属性。基于这个属性的运用，可以将页面内容定位分成静态定位、相对定位、绝对定位、固定定位和浮动 5 种方式。

11.1.1　静态定位

position 默认情况下定义的便是 static 属性，此时的框模型是静态定位，和其他文本内容配合。代码的写法是：

```
position:static;
```

由于属于默认属性，所以通常都省略在代码中写出来。如程序 11.1 静态定位的页面形式。

【本节示例参考：资料光盘\第 11 章\11-1　默认情况下的静态定位.html】

【实例 11-1】默认情况下的静态定位，其源码展示如下：

程序 11.1　默认情况下的静态定位.html

```
01	<html>
02	  <head>
03	    <title>页面内容的定位</title>
04	    <style>
05	      body {font-size:1.5em;              //字体大小
06	            color:white;
07	            text-align:center;             //定义文本居中对齐
```

```
08                              }
09                  #block1 {background:navy;
10                              padding:1em;            //设置空距为 1em
11                              }
12                  #block2 {position:static;           //默认的情况下可以省略
13                              left:20px;              //定义层距离窗口左边 20px
14                              top:20px;               //定义层距离窗口顶部 20px
15                              background-color:orange;
16                              padding:1em;
17                              }
18                  #block3 {background-color:green;
19                              padding:1em;
20                              }
21              </style>
22          </head>
23          <body>
24          <div id="block1">区域 1</div>
25          <div id="block2">区域 2</div>
26          <div id="block3">区域 3</div>
27      </body>
28      </html>
```

【运行程序】浏览该页面，结果如图 11.1 所示。



图 11.1　静态定位



注意：在本例中 static 属性的定位下，代码第 13 和 14 行不起作用，放在本例中是为了说明后面不同属性下位置的样式。

【深入学习】代码第 12 行交代了本例中的层定位，是和系统默认情况下一样的。所以，每一块区域都自然无缝地结合在一起，从上至下，如果设置的是行内对象，那么区域按照从左至右无缝拼接。

11.1.2　相对定位

如果将实例 11-1 中的 position 属性改成 relative，其作用表示相对定位，那么它所相对的参照物，就是 static 属性下的位置，也就是默认情况下的位置。当设定不同的数值时，相对于初始位置发生改变，而初始位置会留下空白占位。这里可以通过 top、right、bottom 和 left 属性来控制位移。如若代码 12 和 13 行属性改写成：

```
position:relative;                    //采用相对位置定位
left:20px;
```

那么，这两句代码表明所约束的对象相对于初始位置向右偏离 20px 的位置。所以当把程序 11.1 中的 position 属性改成 relative 后，其结果如图 11.2 所示。



图 11.2　相对定位

由图 11.2 中可以发现，区域 2 偏离了原先的初始位置（居左 20px，居上 20px）。而偏移之后的初始位置，依然是一片空白。

11.1.3　绝对定位

绝对定位 absolute 是使用最多的属性之一。较之于相对定位 relative，它的特点在于当对象发生位移时，原先初始位置的内容如同被去除了一样，这个对象独立于其他的页面内容，而初始位置的空白会被其他内容自然填补。

此外，绝对定位的框并非是相对于初始位置发生位移。事实上，它是相对于上一级的框的初始位置发生位移。如果上一级的框是浏览器窗口，那么它就是相对于整个页面来发生位移。同样，绝对定位也可以使用 top、right、bottom 和 left 属性来控制位移。

如果将程序 11.1 中的第 12 行和第 13 行中 position 属性和居左距离修改为：

```
position:absolute;
left:20px;
```

那么结果将如图 11.3 所示。结果区域 2 独立于其他页面内容被分离了出来。由于上一级的框即是页面本身，所以，它所发生的位移是相对于浏览器窗口向右偏移 20px，向下偏移 20px。而区域 1 和区

域 3 结合在了一起，就好像从来都没有过区域 2 一样。那么，什么是相对于上一级的框发生位移呢？如程序 11.1 的例子，在样式表的定义中添加一个新的定义，插入的代码是：

```
span {position:relative;
      background-color:black;
       }
```

那么，这个行内标签就是上一级的框模型，而接着将代码第 24 行写为：

```
<span>这里上一级的“框”
   <div id="block2">区域 2
     </div>
</span>
```

所以，这里的区域 2 被放置在了<span>标签内，即放在一个<span>定义的框中，它是 block2 框模型的上一级。结果如图 11.4 所示。

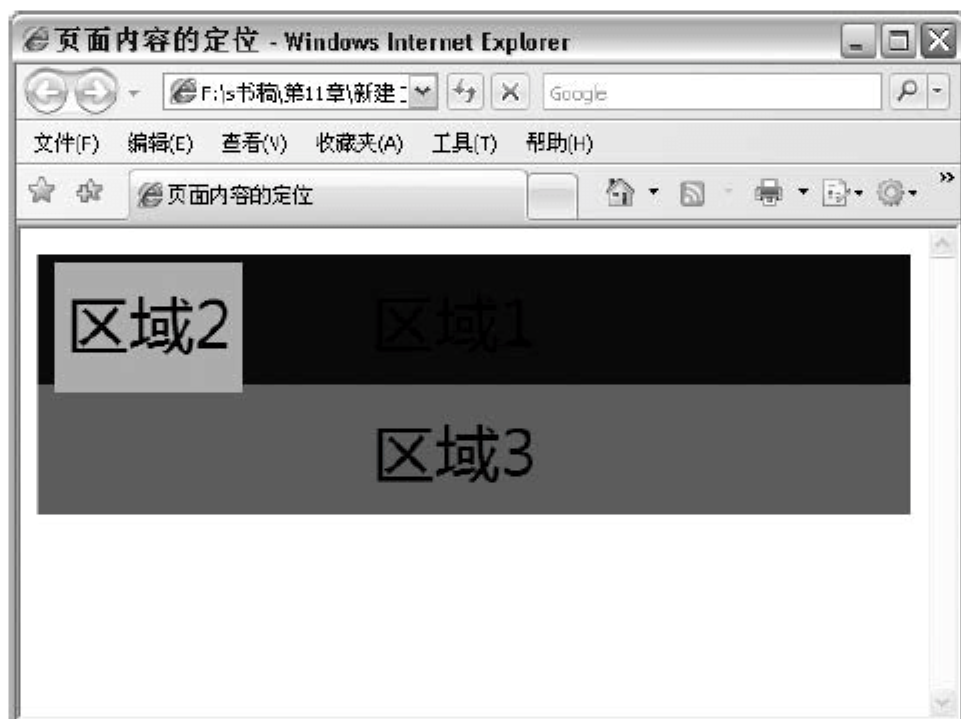


图 11.3　绝对定位

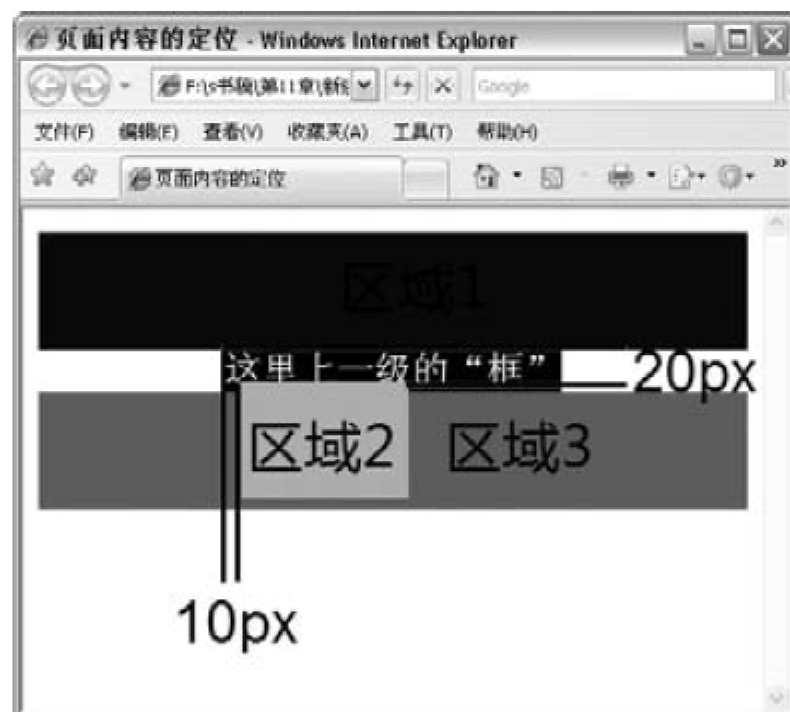


图 11.4　相对于上一级“框”的绝对定位

如图 11.4 所示，区域 2 是相对于它的上一级框，即黑色的这个框所发生的位移，以黑色框的上边向下偏移 20px，向右偏移 10px。

11.1.4　固定定位

固定定位比较类似于绝对定位，当页面长度超出浏览器窗口时，会出现滚动条。区别就在于绝对定位下的页面对象的框，会随着滚动条和页面一起移动，而固定定位下的页面对象的框则不会随之滚动。同样，固定定位也可以使用 top、right、bottom 和 left 属性来控制位移。



注意：IE 7 之前版本的 IE 浏览器不支持固定定位的框。

固定定位和绝对定位的性质是一样的，它们所定义的框的位置是独立于其他页面内容之外的。这样，难免有时它们会叠加在一起，这种情况可以使用 Z 轴属性，即层的叠加特性来改变它们的顺序（参考第 10 章）。

11.2　浮　动　层

浮动层可以将所定义的页面内容方便地放置在页面的左边或右边，通常放入图像时使用这种方法会很方便。事实上，浮动层可应用于任何对象，浮动框的代码写法为：

```
float:left;
```

也可以定义成 right 和 none，具体的使用如程序 11.2 创建浮动层的方法。

【本节示例参考：资料光盘\第 11 章\11-2　创建浮动层.html】

【实例 11-2】创建浮动层的方法，其源码展示如下：

程序 11.2　创建浮动层.html

```
01          <html>
02            <head>
03              <title>创建浮动层</title>
04               <style type="text/css" >
05                    body {font: 80%/1.5  黑体;              //定义页面文本字体
06                           }
07                    h3 {font:1.2em  幼圆;
08                            }
09                    #box {width: 12em;                     //定义这个框模型的宽度
10                           float: left;                     //定义在左侧的浮动层
11                           color: white;                   //设置文本的颜色
12                           background: #060;             //设置背景颜色
13                           padding: 1em;                   //设置空大小
14                           margin: 0;                       //设置边距大小
15                           }
16               </style>
17            </head>
18            <body>
19            <h3>一月一日  领导者必须正直  </h3>
20               <p>组织的精神是自上而下树立起来的
21               </p>
22               <p id="box">摘自：彼得.德鲁克《管理:任务、责任与务实》
23               </p>
24               <p>当考察管理者是否诚信时，人们必定会非常重视他的人品是否正直，这一点必
25               定首先会在管理者的人事任用上体现出来，因为领导者正是通过其正直的人品，才能够实
26               现其领导，领导者也正是通过其正直的人品，才树立了别人效仿的榜样…
27               <p>这一点对企业最高领导层的重要性是毋庸置疑的，因为一个组织的精神是自上而
28        .....
29               </p>
30            </body>
31    </html>
```

【运行程序】浏览该页面，结果如图 11.5 所示。浮动层定义在文本的左边，而文本被挤压在右边。所以，浮动层并不是说会浮动在页面的上方而盖住下面的文本。相反，浮动层更像是可以随意嵌入页面的一个技术。此外，如果并不需要浮动层的左右并排存在页面内容，可以使用“clear”属性来清除页面其他内容。在代码的样式表定义中，加入“clear”属性的声明，如下代码所示：

```
span {clear:left;
    }
```

那么，假如在程序 11.2 第 26 句的位置引用这个声明：

```
现其领导，<span>领导者也正是通过其正直的人品，才树立了别人效仿的榜样，在人品这一点上，人们无法弄虚作假，一个领导者的同事，尤其是他的下属们，只要和领导者共事几周，就会知道他是否正直，他们可以原谅别人的无能，疏忽，缺乏安全感甚至是粗鲁无礼，但是他们却无法宽恕别人的不正直，他们也无法宽恕领导者选用不够正直的人。</span>
```

当设计者将这段文本放入<span>标签内时，页面的结果如图 11.6 所示。

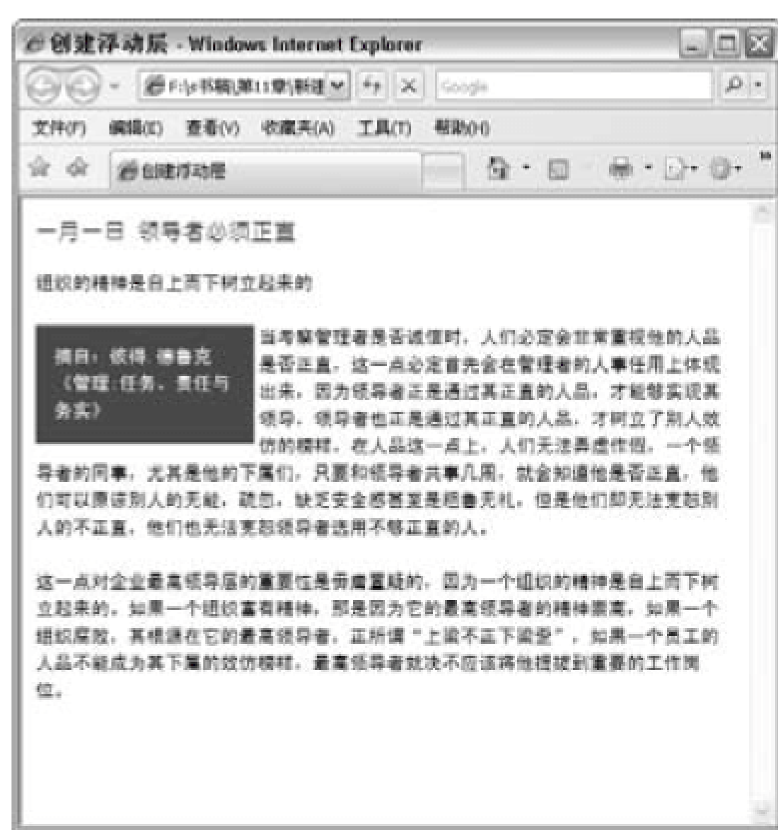


图 11.5　创建浮动层



图 11.6　clear 属性的应用

【深入学习】当使用 clear 属性之后，页面内容将不会和浮动层处于同一行的设定。clear 属性还可以定义为 right 和 both。right 属性定义为在右边不允许浮动元素。both 属性定义为左右两侧均不许浮动元素。

注意：留心对比图 11.5 和图 11.6，可以了解两种不同样式的对比。

11.3　CSS 的新奇技术以及未来发展

在流行的 CSS 样式设计中，包括本书中所讨论的大部分内容都是基于 CSS 2.1。而现在，CSS 3.0 也慢慢地向我们走近，其中多数功能目前还未能被广泛地支持，但是其中不乏一些已经尝试使用的功能。

11.3.1　图像替换技术

图像替换技术是指使用图像替换页面中文本的功能，类似于在页面中插入图像，只是这种方法更为方便，易于代码管理。通常来说，设计者习惯使用有意义的图像去替换一些标题、logo 和某些特定的页面背景。

这里通过一个综合的图像例子，介绍如何在页面中放入图像，设置背景图像，以及使用图像替换技术，如程序 11.3 展示的页面。

【本节示例参考：资料光盘\第 11 章\11-3　使用 CSS 控制图像的技巧.html】

【实例 11-3】使用 CSS 控制图像的技巧，其源码展示如下：

程序 11.3　使用CSS控制图像的技巧.html

```
01      <html >
02        <head>
03          <title>Sweet</title>
04          <style type="text/css" >
05            body {font: 90%/1.5 幼圆;
06                  color: white;
07                  background: black url(图片/dog.jpg) bottom right
08                  fixed repeat-x;                          //设置背景图像
09                  }
10              div { width: 500px;                          //设置层的宽度
11                   margin: auto;
12                  }
13              h1 { width: 280px;
14                   height: 155px;
15                   background-image: url(图片/logo.png);   //文本标题的背景
16                   text-indent: -9999em;                   //将标题位置设定在窗口之外
17                   margin: 0;
18                   }
19              img {                                        //定义图像的大小和边框样式
20                    width: 500px;
21                    height: 150px;
22                    border: 1px solid white;               //设置边框样式
23                    }
24                  p {padding:1em;
25                    width: 500px;
26                    }
27          </style>
28        </head>
29        <body>
30          <div>
31            <h1>Sweet Rain</h1>
32            <p>黑衣服、黑领带、白手套，跟着一只会说话的黑狗，这是"死神"千叶的标准装束。
```

```
33          ......
34          </p>
35              <p>当死神作出不同于以往的判断时，超脱时空的命运之轮开始了转动……</p>
36              <img src="图片/dog.png" alt="sweet rain" />        <!—放入图像替换文本
37              <p>裁断生死的死神因一念之差放了一名女子一条生路，没想到就此引起连锁反应，
38          ......
39          </p>
40          </div>
41          </body>
42          </html>
```

【运行程序】浏览该页面，结果如图 11.7 所示。



图 11.7　使用 CSS 控制图像的技巧

【深入学习】在这个例子中，使用了 3 张图像，分别是“logo.png”、“dog.png”和“dog.jpg”。页面背景图像使用了 dog.jpg。在“body”样式表中，即全局控制情况下，设置背景图像，“fixed”表示将背景图像固定在页面中，不随着页面内容滚动，“repeat-x”表示在横向的方向上重复，所以如果将页面宽度拉长，会出现背景图像的重复现象，如果想避免这种情况，可以通过修改图像本身来实现。



说明：fixed 的完整写法应该是 background-attachment:fixed;。

在第 19～23 行代码中，设置的 img 样式表用来控制插入图像的大小和边框的属性。在代码第 36 行中插入了 dog.png 这张图像。

本例中关键的部分是如何使用图像替换文本，可以看到，代码第 31 行中<h1>标签内本应该是文本 Sweet Rain，注意代码第 14～18 行这个样式表，这个样式表 h1 定义的框模型，表明背景图像 logo.png，事实上，在最终的效果中，确实是一张图像，那么原来的文本是如何消失的呢？其实，文本并没有消

失，留意第 16 行的代码“text-indent: -9999em;”，表示文本框内的文本缩进-9999em 这样的一个值，如此大的数值作用便是将文本推出窗口之外。所以，最终的效果中浏览者并不能看到文本。

11.3.2　CSS 3.0 中的新发展

最近 W3C 组织起草了新的 CSS 3.0 标准（截止于 2008-9-10）。从中可以发现，在新的标准中，大幅扩展了 background 和 border 属性的功能。主要表现在对于背景图像和边框的精细修改，如修改边框的 border-radius，它能使直角边框改成圆角，其效果如图 11.8 所示。定义框模型下的文本阴影效果的 box-shadow 属性，如图 11.9 所示。

border-radius: 55px 25px

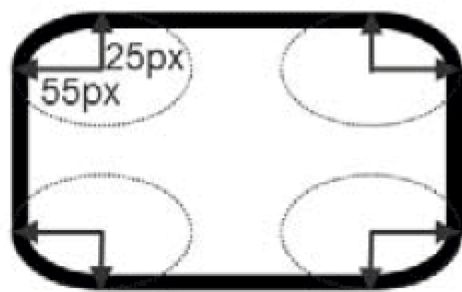


图 11.8　CSS 3.0 中的 border-radius 属性

```
span{border:thin solid;
        box-shadow:0.2em 0.2em #ccc;
        }
```

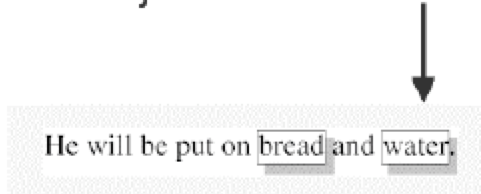


图 11.9　CSS 3.0 中的 box-shadow 属性



说明：资料来自于 W3C 起草方案中的两个示例，如读者有兴趣了解更多，可参考 http://www.w3.org/TR/css3-background/。

技术总是在不断进步、不断发展的，但所有技术的关键核心问题都是基于对原理的理解。这样，在面对所有的扩展的知识上，都能够做到游刃有余、快速上手。

11.3.3　Firefox 浏览器中实现圆角框模型

虽然目前 IE 7.0 还未能有效支持 border-radius 属性，相信在不久的将来可以在 IE 浏览器中很好地实现。目前，在 Firefox 浏览器中已经可以通过特殊的声明来实现这种技术了。这些以-moz 开头的代码专门适用于 Firefox 浏览器的属性声明，称之为 Moz-css。这有时也是一种针对 Firefox 浏览器的 Hack 写法。其中就有-moz-border-radius，它的作用和 border-radius 一样，如程序 11.4 的页面在 Firefox 浏览器下的情况。

【本节示例参考：资料光盘\第 11 章\11-4　Firefox 情况下的圆角边框.html】

【实例 11-4】Firefox 情况下的圆角边框，其源码展示如下：

程序 11.4　Firefox情况下的圆角边框.html

```
01    <html>
02     <head>
03       <title>Firefox 情况下的圆角边框</title>
04       <style type="text/css">
05         div { background-color:maroon;
06              -moz-border-radius: 55px 15px;          //修改边框的圆角
```

```
07                  border: 4px solid black;                //设定边框的样式
08                  color:white;
09                  padding: 30px;                          //设置页面的空距
10                  }
11              </style>
12          </head>
13              <body>
14              <div>在驾车穿越法国乡间时，塞斯·高汀看到一群又一群好像从童话书里出来的奶牛。
15  但是，……
16                  </div>
17                  </body>
18          </html>
```

【运行程序】浏览该页面，结果如图 11.10 所示。



图 11.10　Firefox 情况下的圆角边框

这段代码放在 IE 浏览器中是无法看到圆角效果的。也只有 Firefox 浏览器能解析第 6 行代码。

11.4　案例：有效地管理页面布局

标准化工作当然表现为更为专业的设计习惯，本节将介绍一种 YAHOO 公司的 YUI Grids CSS 推广的布局思路。这是一种很好的布局页面的思路，通过这些可以学习到出色的布局经验，先解决什么问题，再解决什么问题。

（1）设定页面的宽度，设计者可以自由地确定页面的宽度。通过使用 width 属性来固定准确的页面值，或者通过百分比来确定页面。如果使用数值，通常设置在 900px 左右，尽量不要低于 700px，那样页面的主体就显得过于窄小了。如果是通过百分比显示，设置为 100%。代码如下：

```
<style>
#doc {width:750px;
        }
#doc 2{width:950px;
        }
#doc 3{width:100%
        }
```

```
#doc4 {width=974px;
        }
</style>
…
<div id="doc">                //#doc 是 750px 宽的页面
</div>
<div id="doc2">               // #doc2 是 950px 宽的页面
</div>
<div id="doc3">               // #doc3 是 100%填满窗口的页面
</div>
<div id="doc4">               // #doc4 是 974px 宽的页面
</div>
```

如果考虑到在不同浏览器中的显示结果，那么，最好的办法是使用以 em 为单位来设置页面宽度。由于不同的浏览者也许会设置不同的字体大小，这样做的好处是主动去适应浏览者。举一个简单的例子，样式表如果写为：

```
#custom-doc { margin: auto;
              width:46.15em;                    //在非 IE 浏览器中的宽度
              *width:45.00em;                   //在 IE 浏览器中的宽度
              min-width:600px;                  //可以省略，推荐使用
            }
```

在非 IE 浏览器中如果设置页面宽度为 46.15em，那么这个长度相当于在 IE 中设置页面宽度为 45em。所以，这是一种 CSS Hack 的用法，为了避免在不同的浏览器中所见的页面长度大小不一。此外，margin=auto 可以避免页面内容始终贴着浏览器的左侧。

（2）接着将页面的布局分为 3 个部分。一个作为页面的头部，一个作为页面的主体，一个作为页面的底部。相对于这 3 个部分设计 CSS 样式表，分别命名样式表为 header、body 和 footer，代码写为：

```
<div id="doc">
    <div id="hd">          //header
    </div>
    <div id="bd">          //body
    </div>
    <div id="ft">          //footer
    </div>
</div>
```

（3）在页面的主体中，可以进一步细分主体页面的布局，如果把 body 部分分成两个部分，分为 main-block 和 second-block。原理上设置好 second-block 的长度，而 main-block 将使用 second-block 剩余的宽度。代码如下所示：

```
...
 <div id="bd">
     <div id="yui-main">                    //yui-main 对页面不起任何作用
    <div class="yui-b">first</div>          //页面的 first 部分
     </div>
```

```
    <div class="yui-b">second</div>                //页面的 second 部分
</div>
...
```



注意：在这里，yui-main 样式表不具备实际意义，它是一个未定义的样式表，就是说它并不能影响设计者的页面效果。但是，YUI 这么做的意义在于，令 second 内容比 first 内容更容易被搜索引擎优化。

（4）设计者基于这样的基础，可以考虑作出进一步的细分布局。将 main 部分的布局继续一分为二，那么使用这种技术可以很容易地做到内容的嵌套。定义一个 yui-g 的样式表嵌套在 yui-b 中，接着把这部分内容一拆为二，将其中一部分定义为 yui-u，所以，此时代码变成：

```
...
<div id="yui-main">
    <div class="yui-b">
        <div class="yui-g">
            <div class="yui-u first"></div>
            <div class="yui-u"></div>
        </div>
    </div>
</div>
...
```



注意：在 class=yui-u first 这个定义中，使用了伪类:first-child，但并不是所有浏览器都支持这个属性，这里的作用是给 yui-u 添加额外的属性，来区分两个 yui-u 的性质，而带来的好处是可以使用浮动层来划分两边。

最后，设计者可以使用这种方法不停地嵌套下去，但是要注意，如果嵌套的子布局就是其布局的本身，它不需要放在一个嵌套中，如以上代码中的 class=yui-g。而一旦划分出来两个部分，一定要区分标注其中的一个为 first，如以上代码再拆分一次的话，代码是：

```
...
<div id="yui-main">
    <div class="yui-b">
        <div class="yui-g">
            <div class="yui-g first">              ← 在原来的基础上继续拆分一次
                <div class="yui-u first"></div>
                <div class="yui-u"></div>
            </div>
            <div class="yui-g">
                <div class="yui-u first"></div>
                <div class="yui-u"></div>
            </div>
```

```
        </div>
    </div>
</div>
…
```

这一次将 yui-g 分为了两部分来操作，如此反复，可以一直循环下去，YAHOO 公司基于这种原理，开发出一种便捷的 CSS 样式表布局的页面生成工具 YUI:CSS Grid Builder，有兴趣的读者可以去尝试使用。



说明：YUI:CSS Grid Builder 的网址是 http://developer.yahoo.com/yui/grids/builder/。

最后通过实例 11-5 来感受一下这种方法。

【本节示例参考：资料光盘\第 11 章\11-5　管理你的页面布局.html】

【实例 11-5】管理你的页面布局，其源码展示如下：

程序 11.5　管理你的页面布局.html

```
01        <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
02        "http://www.w3.org/TR/html4/strict.dtd">
03        <html >
04        <head>
05          <title>管理页面布局</title>
06          <style type=text/css>
07          html{color: maroon;
08                background: white;
09               }
10          body {font:16px/1.23 幼圆;
11                *font-size:small;                    //设置字体为小号
12                text-align:center;
13               }
14          #doc{margin:auto;                          //使页面能够居中显示
15                text-align:left;
16                width:57.69em;                       //非 IE 浏览器下的页面宽度
17                *width:56.25em;                      // IE 浏览器下的页面宽度
18                min-width:750px;.
19               }
20          .bbb{float:right;
21                color:black;
22               }
23           div.first{ float:left;
24                    width:74.2%;                     //页面主体位置的 74.2%
25                     background:silver;
26                     font:1em 幼圆 ;
27                     color:black;
```

```
28                      padding:10px;
29                      }
30          #bd   {zoom:1;                            //不放大对象的大小和比例
31                    }
32            #ft  { background:teal;                 //#ft 是页面的底部层
33                    font:1em  幼圆 ;
34                    color:orange;
35                    padding:2px;
36                    }
37         </style>
38       </head>
39       <body>
40       <div id="doc">
41             <div id="hd">
42                   <p>他不像精神病或一般小说上所记载的其他多重人格病患一样使用杜撰的假名,
43       ......
44       </p>
45             </div>
46             <div id="bd">
47                   <div class="main">                   //不具备实际意义的#main
48                         <div class="bbb first">
49                               <p>初次见到这位廿三岁的年轻人,是在俄亥俄州雅典市的雅典心理健康中心,
50       ......
51       </p>
52                         </div>
53                              <div class="bbb">
54                               <p>几天之后,《新闻周刊》一篇名为《比利的十个面孔》文章最
55             ......
56             </p>
57                               </div>
58                         </div>
59                   </div>
60             <div id="ft">
61                         <p>然后有一天,令人惊异的事情发生了。威廉. 密里根第一次完全融合
62             ......
63             </p>
64                   </div>
65              </div>
66             </body>
67          </html>
```

【**运行程序**】浏览该页面,最终的结果如图 11.11 所示。



图 11.11　管理你的页面布局



注意：代码第 30 和 31 行，zoom 属性用来触发 layout 属性，layout 是一个 IE/Win 的私有概念，它决定了一个元素如何显示以及约束其包含的内容，如何与其他元素交互和建立联系，如何响应和传递应用程序事件/用户事件等。zoom:1 表示就是不放大对象的大小和比例。

此外，在布局页面时，要养成一个良好的定义样式表命名的习惯，如表 11.1 所示。

表 11.1　样式表命名

页面内容的位置	命 名 习 惯
容器	container
页头	header
内容	content
页面主体	main
页尾	footer
左右中	left right center
导航	navigation
栏目	column
侧栏	sidebar
页面外围控制整体布局宽度	wrapper
主导航	mainbav
子导航	subnav
左导航	leftsidebar
顶导航	topnav
右导航	rightsidebar
底导航	bottomnav
标题	title
摘要	summary
菜单	menu

续表

页面内容的位置	命 名 习 惯
子菜单	submenu
边导航	sidebar
标志	logo
登录	login
登录条	loginbar
注册	regsiter
加入	joinus
功能区	shop
搜索	search
状态	status
按钮	button
标签页	tab
文章列表	list
提示信息	msg
小技巧	tips
投票	vote
指南	guild
广告	banner
热点	hot
注释	note
图标	icon
合作伙伴	partner
友情链接	flink
版权	copyright
下载	download

11.5 小 结

在第 10 章的基础上，本章介绍了如何职业化地管理好你的页面布局，CSS+DIV 是一种流行布局方式，不仅是一项时髦的技术，更是一门值得深究的学问。在了解的基础上，只有熟练才能生巧。当能够自由地在窗口中布局你的页面时，CSS 会带给你极大的成就感和乐趣。本章主要的知识点有：

- ❑ 框模型的定位，有静态定位、相对定位、绝对定位和固定定位。
- ❑ 浮动层的使用以及 clear 属性。
- ❑ 在页面中使用图像，以及图像替换技术。
- ❑ CSS 3.0 的未来发展中的奇特技术，如圆角、阴影。
- ❑ YUI Grid CSS 的重复嵌套的布局方法。

在接下来的章节中，将开始接触页面中的脚本程序，如果说目前的网站前端开发主力技术还是 CSS+DIV，那么今后的页面开发是属于 CSS+DIV+JavaScript 的。

第 3 篇　动感页面篇



第 12 章　神奇的表单

本章将真正地接触 DHTML，之前的所有网页浏览者只能看到，或者使用超链接去浏览另一个页面或页面内的锚点，但是这些却不能算是动态页面。现在，大多数网站都具备收集用户信息的功能，如发表留言、输入账号等，这些动态功能都是通过表单来实现的。本章的主要知识点如下。

- 表单是如何工作的。
- 如何创建表单。
- 不同功能多种样式的表单。
- 表单域能够做些什么。

前面了解了什么是 JavaScript 程序，而事实上，要分清两个概念，一个是什么是 JavaScript 的程序部分，另一个是该如何去触发 JavaScript 程序。下面通过一个简单的例子来了解一下，如程序 12.1 所示。在这个例子中，可以了解 JavaScript 程序和表单之间的工作关系以及它们是如何合作的。

【本节示例参考：资料光盘\第 12 章\12-1　计算矩形的面积.html】

【实例 12-1】计算矩形的面积，其源码展示如下：

程序 12.1　计算矩形的面积.html

```
01          <html>
02            <head>
03              <title>计算矩形的面积</title>
04              <style type=text/css>
05                  .result { font-weight: bold;
06                          }
07              </style>
08              <script language="JavaScript">
09               /*
10                * 这是一个计算矩形面积和体积的 JavaScript。
11                * 定义了一个函数作为事件。
12                */
13              function calculate() {
14                  /* 获取用户输入的数据*/
15                  var length = document.loandata.length.value;
16                  var width = document.loandata.width.value;
17                  var height = document.loandata.height.value;
18                 /* 获取<span>标签的对象*/
19                  var area = document.getElementById("area");
20                 /* 输出计算的结果*/
21                  area.innerHTML = length * width;
22                  volume.innerHTML = length * width * height;
```

```
23                                          }
24                 </script>
25              </head>
26              <body>
27                 <form name="loandata">
28                    <table>
29                     <tr><td><b>输入长宽高的数值：</b></td></tr>
30                     <tr>
31                       <td>1) 矩形的长度是:</td>
32                       <td><input type="text" name="length"></td>
33                     </tr>
34                     <tr>
35                       <td>2) 矩形的宽度是:</td>
36                       <td><input type="text" name="width"></td>
37                     </tr>
38                     <tr>
39                       <td>3) 矩形的高度是:</td>
40                       <td><input type="text" name="height"></td>
41                     </tr>
42                     <tr><td></td>
43                       <td><input type="button" value="运行" onclick="calculate();"></td>
44                     </tr>
45                     <tr><td><b>矩形的面积和体积分别是：</b></td></tr>
46                     <tr>
47                       <td> 矩形的面积：</td>
48                       <td><span class="result" id="area"></span></td>
49                     </tr>
50                     <tr>
51                       <td> 矩形的体积：</td>
52                       <td><span class="result" id="volume"></span></td>
53                     </tr>
54                    </table>
55                 </form>
56              </body>
57           </html>
```

【运行程序】浏览该页面，结果如图 12.1 所示。

【深入学习】代码第 8～24 行这部分是一个 JavaScript 的小程序。它的作用是计算一个矩形的面积和体积。那么，对于页面浏览者来说，如何才能使用这个计算矩形的小程序呢？这里就要通过表单将小程序触发，并且通过表单使这个程序可以和用户交互。

从实际浏览结果来看（如图 12.1 所示），其中可以由用户输入数据的功能便是表单的作用。这部分代码是第 32、36、40、43、48、52 这 6 行。所以，当用户输入长度数值为 3、宽度数值为 4、高度数值为 5 这样的数值时，到这里都是表单在起作用。而页面中的小程序只负责计算出矩形的面积和体积。

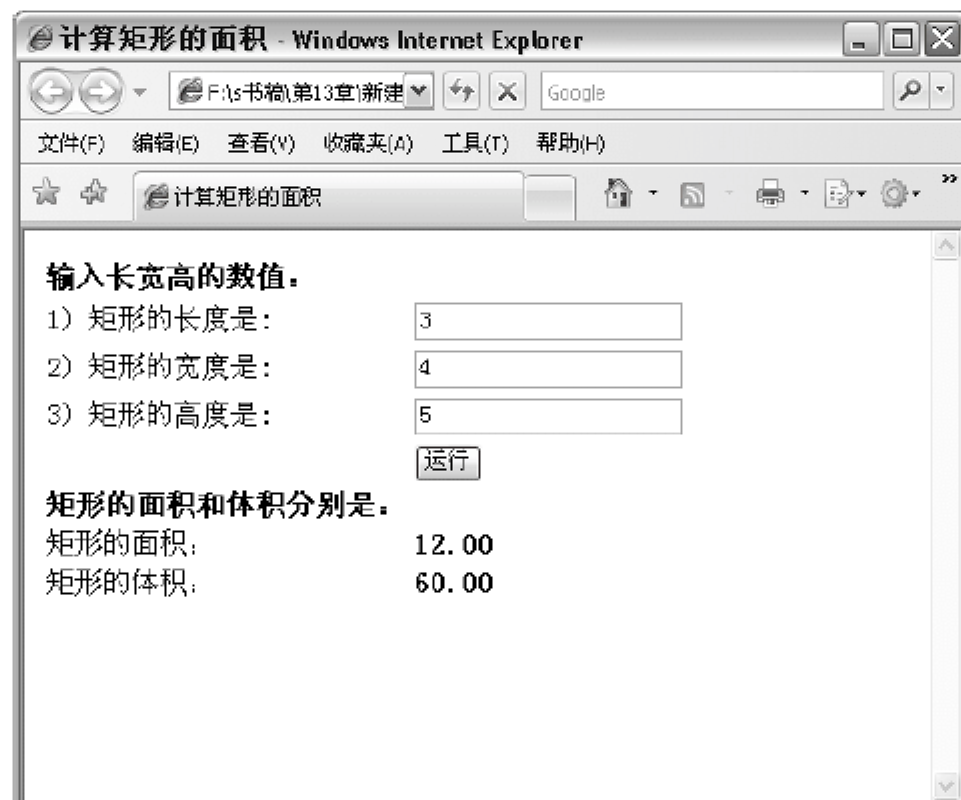


图 12.1　计算矩形的面积



注意：在本章中，重点讨论如何使用表单来触发 JavaScript 的程序，如何配合设计者来表现页面中的表单。而什么是 JavaScript 程序将在后面的章节中介绍。

12.1　表单的工作原理

表单最重要的表现就是在客户端接收用户的信息，然后将数据递交给后台的程序来操控这些数据。从技术的概念上来说，表单就是用来操作 form 对象，对象是一种基本的数据类型。

12.1.1　<script>标记

JavaScript 程序的调用类似于 CSS 样式表，可以像嵌入式样式表一样放在<head>标签中，也可以像外联样式表一样通过链接来引用。所以，当放到<head>标签中使用时，需要通过<script>标记命令行，如程序 12.1 中第 8 行和第 24 行。浏览器通过<script>标签获得分析程序的信息，来告诉浏览器使用的是哪种语言的脚本，如程序 12.1 中第 8 行。

```
<script language="JavaScript">
```

告诉浏览器，这是 JavaScript 脚本。

如果是通过引用外部 JavaScript 程序，就像链接外联样式表那样，那么代码应该写成：

```
<script type="text/javascript" src="some.js">
</scirpt>
```



说明：JavaScript 程序的后缀名是.js。JavaScript 的程序设计参考第 13、14 章。

12.1.2　创建表单

创建一个表单看上去就像创建表格，表格的单元格、行、列都放在<table>标签中，而表单通过<form>标签来创建，其中放置表单的对象，如表单域、按钮和其他事物。

<form>标签中可扩展几个属性，分别是 action、method、name、enctype 和 target 属性。action 属性表示将数据传送到指定的 URL 地址，method 属性的值告诉浏览器通过怎样的方式来提交数据，name 属性保证了表单数据能够被程序识别。还有 enctype 属性和 target 属性，前者用来表示编码方式，后者用来表示目标的显示方式。

1．action 属性

通过<form>标签定义的表单，它里面必须有 action 属性才能将表单中的数据提交出去。如这句代码所示：

```
<form action="some.php">
</form>
```

它表明的是这个表单的作用是用来提交 some.php 这个页面中的数据。

2．method 属性

method 属性的作用是告诉浏览器数据是以何种方式提交出去的。method 属性下可以有两个选择：post 或者 get。默认情况下，数据被提交的方式是 get，表单域中输入的内容会添加在 action 指定的 URL 中。当表单提交之后，用户便获得一个明确的 URL。由于这种方式下数据会添加到 URL 中，所以，可利用的一个好处是可以保存在收藏夹中和他人分享，直接访问主页的 URL 可以达到和填写注册后一样的效果。如有些时候，用户将自己已经注册过的一些网站主页加入到自己的收藏夹，再次从收藏夹中打开时，会发现已经是登录的状态。而 post 这种方式，数据将以 HTTP 头的形式发送，表单数据不再是 URL 中的一部分。

这两种方式的区别在于，get 在安全性上较差，所有表单域的值直接呈现。而 post 除了可见的处理脚本程序以外，别的东西都可以隐藏。所以在实际运用时，通常选择 post 这种处理方式。

3．name 属性

添加 name 属性是为了令递交出去的表单数据能够被处理这些数据的程序识别。在大部分页面中，很可能放入的表单不止一个，那么就要通过给不同的表单起不同的名字，便于程序来识别，如程序 12.1 中的第 27 行。

```
<form name="loandata">
```

这里将表单命名为 loandata，而在第 15 行：

```
var length = document.loandata.length.value;
```

表示通过表单 loandata 获取输入的 length 项数值。如果代码中有很多不同的表单，那么通过 name 就可以区分它们。

4．编码方式

enctype 代表 HTML 表单数据的编码方式，分别有 application/x-www-form-urlencoded、multipart/form-data

和 text/plain 共 3 种方式。application/x-www-form-urlencoded 是标准的编码方式，提交的数据被编码为名称/值对。multipart/form-data 属性表示数据编码为一条信息，为表单定义了 MIME 编码方式，创建了一个与传统不同的 POST 缓冲区，页面上每个控件对应消息中的一个部分。text/plain 表示数据以纯文本的形式进行编码，这样在信息中将不包含控件或格式字符。



注意：multipart/form-data 方式上传文件时，不能使用 post 属性。

5．目标显示方式

表示在何处打开目标 URL。可以设置 4 种方式，_blank 表示在新的页面中打开链接，_self 表示在相同的窗口中打开页面，_parent 表示在父级窗口中打开页面，_top 表示将页面载入到包含该链接的窗口，取代任何当前在窗口中的页面。

所以，一个完整的<from>标记如下所示：

```
<from action="mailto:depp59@gmail.com"
        method="post"
        name="some"
        enctype="text/plain"
        "target="_blank">
…
</form>
```

这段代码表明这个表单的动作是发送到邮箱 depp59@gmail.com，使用 post 的传输方式，使用 text/plain 编码方式的 some 表单，使之在新页面中打开。

12.1.3 表单域

表单域是用户输入数据的地方。说得形象一些，就相当于是用户给程序下命令或者给指示的聊天窗口。当然，这种下命令的方式有许多，如最常见的文本域（参考第 2 章中实例 2-1，其中就包含了一个简单的文本域表单）、下拉列表等。表单域可分为 input、textarea 和 select 3 个对象，其中，大部分类型的表单形式都通过 input 属性来实现，textarea 和 select 创建一种控制类型。在 12.2 节中，将重点分析依赖这些对象实现的表单类型。

12.2 通过表单展示不一样的页面

表单中包含多种不同样式、不同功能的提交数据的方式。在许多页面中，浏览者不经意间已经不断地在使用表单的功能，如留言、设置自己的密码，或者复选框、下拉列表等。不同的功能实现不同的作用。

12.2.1　input 对象下的多种表单表现形式

通常，在页面中见到的大部分表单的形式都是通过输入标记 input 来实现的，一个简单的样式看上去可以是这样的：

```
<input name=""
       type=""
       value=""
       size=""
       maxlength="" >
```

- name：表示输入数据的名字，它的作用也是为了让程序明白所提交的数据，如程序 12.1 中第 32 行。

```
<input type="text" name="length">
```

这个输入的数据被命名为 length，在第 15 行中：

```
var length = document.loandata.length.value;
```

前面的一个 length 是程序定义的标识符为 length，后面的一个 length 则是表示获取通过第 32 行提交的 length 数值。如果第 32 行中缺少了这样一个 name 属性，虽然在浏览器中显示没有什么变化，但事实上，后台程序或 JavaScript 程序就不能获得提交的数据。

- type：表示所定义的是哪种类型的表单形式，该属性有 9 个可选值，每个类型都有自己的功能，如表 12.1 所示。

表 12.1　样式表命名

名　　称	功　　能
text	单行的文本框
password	将文本替换为"*"的文本框
checkbox	只能做二选一的是或否选择
radio	从多个选项中确定的一个文本框
submit	确定命令文本框
hidden	设定不可被浏览用户修改的数据
image	用图片表示的确定符号
file	设置文件上传
button	用来配合客户端脚本

- size：表示文本框字段的长度。
- maxlength：表示可输入的最长的字符数量。
- value：表示预先设置好的信息。

12.2.2　text 文本框的样式表单

text 样式下的文本框是一个单行的文本框，比较常见于"登录"这样的使用，如程序 12.2 是一个有着 text 样式的表单。

【本节示例参考：资料光盘\第 12 章\12-2 text 样式的表单.html】

【实例 12-2】text 样式的表单，其源码展示如下：

程序 12.2 text样式的表单.html

```
01          <html>
02            <head>
03              <style type="text/css">
04            div { width:400px;
05                  font: 180%  微软雅黑;
06                  margin:auto;                          //使页面内容居中
07                  padding:10px;                         //空距是 10px
08                  text-align:right;
09                              }
10          input {font: 50%  微软雅黑;                    //表单内的字体
11                  }
12              </style>
13            </head>
14            <body>
15             <div>
16               <form action="..." method="post">
17                 输入用户名：<input name="name" type="text" size="20" maxlength="12">
18                 <br>输入邮箱地址：<input name="address" type="text" size="20"
19   maxlength="20">                                     //设置放入文本的表单域
20              </form>
21             </div>
22            </body>
23   </html>
```

【运行程序】浏览该页面，结果如图 12.2 所示。

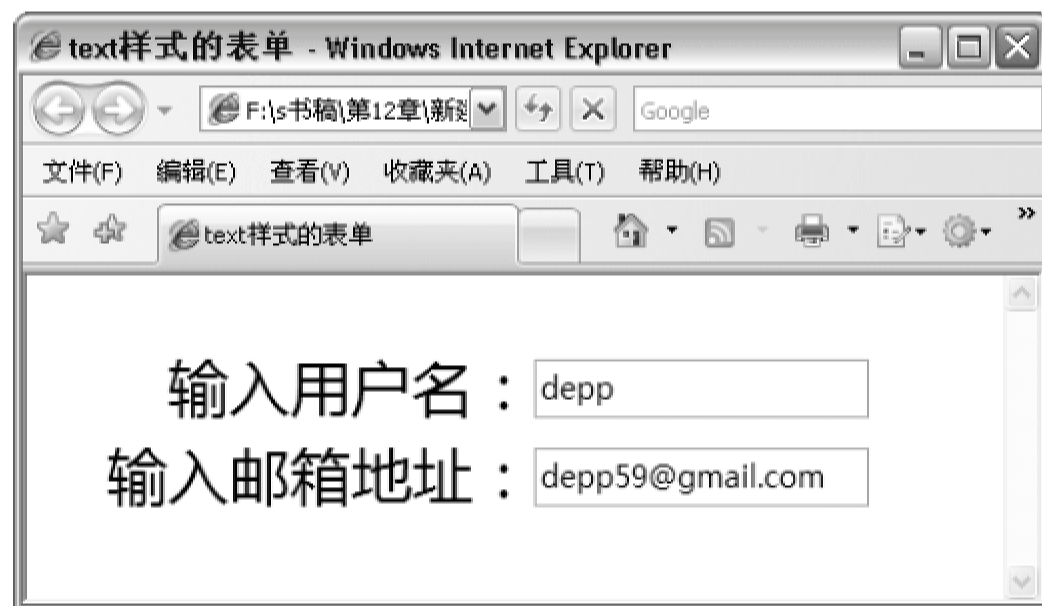


图 12.2 text 样式的表单

【深入学习】注意第 17 和 18 行，其中 size 属性定义了文本框的长度，而 maxlength 属性定义了这个文本框最多只能输入 12 个字符。如果超出这个长度，数据将不能被输入。而这两个 text 样式的数据定义了不同的名字。这很关键，否则一旦需要被程序调用，数据将无法被辨认。此外，可以发现，表单也是可以通过 CSS 样式表来控制的。

12.2.3　password 输入密码的样式表单

这是一个可以将文本使用保密形式展示出来的一个功能，最常见的使用莫过于密码的设置。根据不同的浏览器，会使用点状形态或者星号符来表示，如果在程序 12.2 中表单部分代码写为：

```
<form action="..." method="post">
   输入用户名：<input name="name" type="text" size="20" maxlength="12">
  <br>输入邮箱地址：<input name="address" type="text" size="20" maxlength="20">
  <br>输入密码：<input name="secret" type="password" size="20" maxlength="20">
</form>
```

那么，这段代码的结果显示为如图 12.3 所示。



图 12.3　password 样式的表单

12.2.4　checkbox 复选框的样式表单

checkbox 是一个复选框的创建方式，类似于一个开关的 on 和 off 选项，浏览器会在选择栏前面提供一个方形小框。如果选择符合的意思，小框中会添加小勾符号表示被选中，如程序 12.3 是一个 checkbox 样式的表单。

【本节示例参考：资料光盘\第 12 章\12-3　checkbox 样式的表单.html】

【实例 12-3】checkbox 样式的表单，其源码展示如下：

程序 12.3　checkbox样式的表单.html

```
01          <html>
02             <head>
03                <title>checkbox 样式的表单</title>
04                <style>
05                   body { font: 100%   微软雅黑;
06                         }
07                   #leftblock{position:absolute;               //#leftblock 在页面设置为绝对定位
08                                  width:100px;
09                                  font: 120%   微软雅黑;
10                                  text-align:right;
```

```
11                          }
12              #rightblock {position:absolute;           //#rightblock 在页面设置为绝对定位
13                          width:300px;
14                          left:120px;                   //设置#rightblock 在页面中的位置
15                          padding:5px;
16                          border:2px dotted ;           //设置表单域的边框样式
17                          text-align:left;
18                          }
19          input {font: 50%   微软雅黑;
20                  }
21          h1 {font: 80%   微软雅黑;
22              margin:5; }
23      </style>
24      </head>
25      <body>
26          <div id="leftblock">注册信息:</div>
27          <div id="rightblock">
28              <form action="..." method="post">
29                  <input name="truename" type="checkbox" checked="checked">使用真实姓名
30                      <h1>实名制可以方便您更好地和朋友交流</h1>
31                  <input name="address" type="checkbox" checked="checked">显示我的地址
32                          <!—checked="checked"使复选框默认为选中状态-->
33                      <h1>如果去除勾选，其他用户将无法查到你的地址</h1>
34                  <input name="mail" type="checkbox" checked="checked">可以给我发邮件
35                      <h1>如果勾选，我们将会为你发送来自广告商的信息</h1>
36              </form>
37          </div>
38      </body>
39  </html>
```

【运行程序】浏览该页面，结果如图 12.4 所示。



图 12.4　checkbox 样式的表单

【深入学习】表单的代码是第 28～36 行，在表单中添加 checked="checked"，表示复选框默认值设置为 checked，那么勾符号会被默认添加，如图 12.4 所示。

12.2.5　radio 单选按钮的样式表单

radio 样式有点类似于选择题，在一组选项中选出唯一的一项，发送这列数据。使用的方法上给同一组选项定义为同一名字，但是通过 value 属性来辨识它们之间的不同。具体的使用如程序 12.4 页面 radio 样式的表单。

【本节示例参考：资料光盘\第 12 章\12-4　radio 样式的表单.html】

【实例 12-4】radio 样式的表单，其源码展示如下：

程序 12.4　radio样式的表单.html

```
01    <form action="..." method="post">
02       <input name="onechoice" type="radio" value="one" checked="checked">使用邮箱注册
03                                                         //设置 radio 样式表单的属性
04               <h1>您可以通过自己习惯的邮箱来作为账号登录网站</h1>
05             <input name="onechoice" type="radio" value="two">通过手机注册
06                <h1>您可以通过手机免费获得我们的账号</h1>
07             <input name="onechoice" type="radio" value="three">申请我的 ID 号码
08                <h1>您可以通过网站直接申请账号</h1>
09    </form>
```

【运行程序】将程序 12.4 中第 28～36 行的 form 表单替换为上面的 radio 样式的表单代码，最终页面的显示效果如图 12.5 所示。



图 12.5　radio 样式的表单

【深入学习】radio 样式的表单是一个多选一的表单功能，同样，这里也可以使用 checked="checked" 语句来确定预先选中的一项。当选择唯一的目标后，这个选项将会以数据形式被发送。所以，这里必须给 input 对象设定 value 值，而且不同对象的 value 值也不能相同，否则数据无法被辨认，如这个表单中 value 属性值依次设置的是 one、two 和 three，互不相同。

12.2.6　submit 提交数据的样式表单

submit 属性创建一个按钮，一个按钮的作用就是提交数据。当然准确点说，submit 属性负责“提

交”这样的一个动作。当单击“提交”按钮时，数据会发送到表单指定的地方。如果将程序 12.3 中的 checkbox 表单修改为 submit 表单。那么，submit 的表单样式的写法如程序 12.5 所示。

【本节示例参考：资料光盘\第 12 章\12-5　submit 样式的表单.html】

【实例 12-5】submit 样式的表单，其源码展示如下：

程序 12.5　submit样式的表单.html

```
01      <form action="..." method="post">
02        <input name="onechoice" type="radio" value="one" checked="checked">使用邮箱注册
03          <h1>您可以通过自己习惯的邮箱来作为账号登录网站</h1>
04          <input name="onechoice" type="radio" value="two">通过手机注册
05          <h1>您可以通过手机免费获得我们的账号</h1>
06        <input name="onechoice" type="radio" value="three">申请我的 ID 号码
07            <h1>您可以通过网站直接申请账号</h1>
08        <br>
09            <input type="submit" value="   确  定   ">
10      </form>
```

【运行程序】浏览该页面，结果如图 12.6 所示。

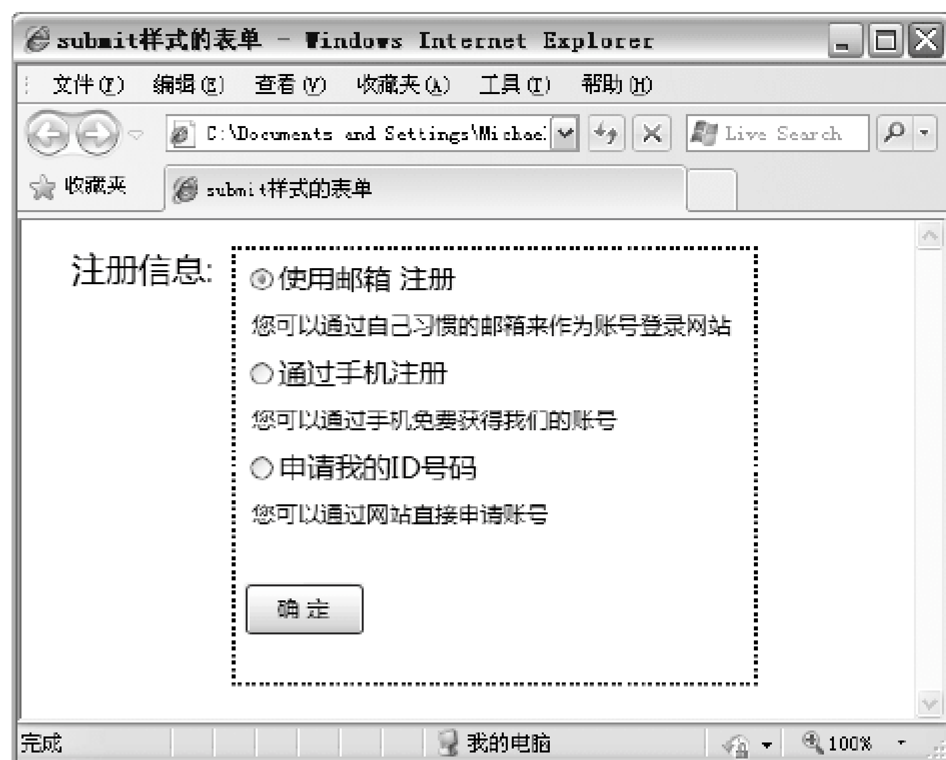


图 12.6　submit 样式的表单

【深入学习】如图 12.6 中所示，有一个醒目的“确定”按钮。它就是通过代码第 9 行来实现的，这也是一个 submit 属性提交表单数据的按钮。其中，通过 value 属性，设计者可以修改按钮上显示的内容。

此外，和 submit 属性类似的还有一个 reset 属性，这是一个复位按钮。当被单击时，表单的内容会被重新设置，回到页面的初始状态。它的代码和 submit 样式类似，代码如下：

```
<input type="reset" value="   恢  复   ">
```



说明：创建 submit 按钮或 reset 按钮时，name 属性不是必需的。

12.2.7　hidden 隐藏域的样式表单

hidden 属性可以创建一个隐藏域，数据会被隐藏起来。因此用户是无法被操作的，这样说来似乎 hidden 没有什么作用。事实上，正是出于安全的考虑，在多步操作数据的同时，用 hidden 来记录页面的数据并将它隐藏起来，这些数据通常是用户并不关心的，但必须是被提交的数据，如它可能是用户操作时的特殊数据，用户并不在意，但必须被提交。

然后当页面跳转到下一个页面的同时，页面已经继承第一个页面的数据，但用户是看不到的。最后，将用户提交的所有数据一并发送至服务器。

通常这种方式运用于动态页面制作，当填写好第一张表单时，处理表单的脚本程序可以动态地生成第二张表单。同时，其中包含了第一张表单的一些数据，如它们可能看上去是这样的：

```
<form action=some.asp>
  <input type=hidden name=somehidden value=some>
  <input type=submit value=下一页>
</form>
```

当单击“下一页”按钮，跳转到第二个页面时，页面会记录第一个页面中的数据：<%=request ("somehidden")%>。



注意：通过 HTML 页面源码可以查看该元素属性的值，所以不要使用 hidden 来传送敏感信息，如密码、手机号码等。

12.2.8　image 样式的表单

image 样式的表单看上去就像是在页面中放入图像，或者又有些类似于图像替换文本的技术，那不妨将其看作是用图像替换按钮的技术。当图像被单击时，数据一并被提交至服务器，代码如下：

```
<input type="image" src="图片/小图标.jpg" alt="　确　定　">
```

同样在编辑图像时，使用 src 属性指定这张图像的路径，使用 alt 属性来添加文本注释。它的效果如图 12.7 所示。



图 12.7　image 样式的表单

如图 12.7 所示，提交按钮被替换成一个小小的图像。当单击图像时，其作用就相当于 submit 按钮。不过，当表单数据被提交的同时，用户所单击的图像的位置坐标也会被发送，就像这样：

```
image.x=23
image.y=59
```

不仅仅是使用图像作为按钮，表单中还有一种触发事件的 button 表单。button 样式顾名思义，它只是一个按钮，单个 button 按钮并不会提交任何数据，它的作用是用来调用前端页面，即客户端的脚本程序，如程序 12.1 的第 43 行用来调用一个简单计算的 JavaScript 脚本。

```
<input type="button" value="运行" onclick="calculate();">
```

12.2.9 file 上传文件的样式表单

file 样式的表单允许用户上传自己的文件，这在论坛、社区类型的网站中经常遇到。例如，用户上传自己的图像给服务器，用来改变用户在不同网站上的形象图片。程序 12.6 所示就是 file 样式的表单。

【本节示例参考：资料光盘\第 12 章\12-6 file 样式的表单.html】

【实例 12-6】file 样式的表单，其源码展示如下：

程序 12.6 file样式的表单.html

```
01          <html>
02            <head>
03               <title>file 样式的表单</title>
04               <style type="text/css" >
05               body {font: 120% 微软雅黑;                 //设置页面字体的样式
06                     }
07               input {font:100% 微软雅黑;                  //设置表单中按钮字体的样式
08                          }
09               </style>
10            </head>
11            <body>
12               上传我的文件：
13              <form action="..." method="post" enctype="multipart/form-data">
14                 <input type="file" name="uploadfile" id="uploadfile" />
15              </form>
16            </body>
17          </html>
```

【运行程序】浏览该页面，结果如图 12.8 所示。

【深入学习】需要注意的是，当使用 file 样式的表单时，必须在 form 标签中说明编码方式，如 enctype="multipart/form-data"。这样，服务器才能接收到正确的信息。



图 12.8　file 样式的表单

12.3　textarea 对象的表单

textarea 对象就像是 input 对象中 text 样式的表单，只不过是扩展过的 text 样式表单，可以通过行（rows）属性和列（cols）属性来编辑文本域的大小，最常见于留言板、论坛中回帖时的文本框等，如程序 12.7 的 textarea 对象的表单。

【本节示例参考：资料光盘\第 12 章\12-7　textarea 对象的表单.html】

【实例 12-7】 textarea 对象的表单，其源码展示如下：

程序 12.7　textarea对象的表单.html

```
01          <html>
02            <head>
03               <title>textarea 对象的表单</title>
04               <style type="text/css" >
05               body {font: 120%  微软雅黑;        //设置页面字体的样式
06                         }
07               textarea {font:80%  微软雅黑;
08                                color:navy;                  //设置文本域字体颜色
09                                   }
10               </style>
11            </head>
12            <body>
13               留言板
14              <form action="..." method="post" enctype="multipart/form-data">
15                <textarea name="some" rows="10" cols="50"   value="say"> 请文明用语:
16                </textarea>
17              </form>
18            </body>
19          </html>
```

【运行程序】 浏览该页面，结果如图 12.9 所示。



图 12.9　textarea 对象创建的留言板

【深入学习】textarea 属性标签是必须要封闭的，如代码第 15 行和第 16 行。此外，在<textarea>标签中放入文本，如代码第 15 行中“请文明用语：”，那么在生成页面时，会预先设置好文本，它可以给用户带来亲切的感受，但同时，用户不得不先删除预先的文本再编辑自己的内容，所以如何取舍就要因地制宜了。



说明：当在文本框中输入的内容超出预先设置的行数时，会自动出现滚动条。如果没有超出文本框的范围，滚动条呈灰色状。

12.4　select 对象的表单

select 对象的表单将创建出一个列表样式的表单，显示为出现一个下拉列表框，令用户可以方便地选择其中一个目录。通常，在一些要求户口填写地区、生日等信息中，设计者可以给使用者准备好选项，为了令使用者填写信息时方便，在代码的写法中需要使用<option>标签来定义可供选择的每一项，如程序 12.8 页面中 select 对象的表单。

【本节示例参考：资料光盘\第 12 章\12-8　select 对象的表单.html】

【实例 12-8】select 对象的表单，其源码展示如下：

程序 12.8　select对象的表单.html

```
01          <html>
02            <head>
03               <title>select 对象的表单</title>
04               <style type="text/css" media="all">
05                      body {font: 120%  微软雅黑;
06                                  }
07                      select {font:80%  微软雅黑;
08                              color:red;                   //select 对象的表单中文本字体颜色
09                              }
10               </style>
```

```
11          </head>
12          <body>
13              <form action="">
14                地址：
15              <select name="上海" >
16                <option>黄浦区</option>                  //这里是提供的每个选项的内容
17                <option>虹口区</option>
18                <option>静安区</option>
19                <option>长宁区</option>
20                <option>杨浦区</option>
21                <option>宝山区</option>
22                <option>浦东新区</option>
23                <option>徐汇区</option>
24                 <option>普陀区</option>
25              </select>
26              </form>
27            </body>
28          </html>
```

【运行程序】浏览该页面，结果如图 12.10 所示。



图 12.10　select 对象的表单

【深入学习】用户可以通过下拉列表框选择一个“地址”，而这个数据则会被表单发送到服务器。此外，还可以使用 value 属性为每一个 option 指定不同的值。如果是这样，value 设置的值将取代 option 的文本内容。



注意：如果设计者希望预先设置初始值，那么在所希望的 option 中添加 selected="selected" 就可以了，如<option selected="selected">浦东新区</option>。

此外，如果下拉列表框中的选项太多，可以使用<optgroup>标签配合 label 属性来给选项分类，如将代码第 15～25 行的代码修改为如下代码：

```
01              <select name="上海" >
02                <optgroup label="Team1">               //给选项分组
03                   <option>黄浦区</option>
```

```
04                <option>虹口区</option>
05                <option>静安区</option>
06                <option>长宁区</option>
07            </optgroup>                       //给选项分组
08            <optgroup label="Team2">
09                <option>杨浦区</option>
10                <option>宝山区</option>
11                <option>浦东新区</option>
12                <option>徐汇区</option>
13                <option>普陀区</option>
14            </optgroup>
15        </select>
```

【运行程序】在上述代码的第 2～7 行、第 8～14 行，通过<optgroup>标签将下拉列表框分割成了两部分，那么在页面中的效果是如何呢？如图 12.11 所示。



图 12.11　<optgroup>标签分组 select 对象的选项

【深入学习】此外，如果设计者不希望 select 对象以下拉列表框的形式展示出来，有一种方式可以将目录项以滚动条的框体样式表现出来。只需要在 select 标签中加入 size 属性，如 size="6"，则表示是一个能容纳 6 行文字的文本框。当目录项超出设置的行数时，将出现滚动条。所以，如果把程序 12.8 中第 15 行写成：

```
<select name="上海"   size="6">
```

那么，页面将变成图 12.12 所示的样子。



图 12.12　select 对象中的 size 属性效果

12.5　表单域集合

如果一个页面中表单的项目过于繁多，设计者可以通过使用表单域将表单分组。当然，表单域未必是只有太长表单的情况下才适合使用。事实上，很多时候，设计者拿这样的方式来修饰表单部分。

表单域的代码是由<fieldset>标签和<legend>标签组合而成的。默认情况下，<fieldset>标签勾画出表单域的框形，<legend>标签的对象像标题一样出现在框形的左上角。代码看上去可能是这样的：

```
<form action="..." method="post">
  <fieldset>
    <legend>注册信息：</legend>
       输入用户名：<input name="name" type="text" size="20" maxlength="12">
       <!—这里可以放入许多样式的表单 --!>
    </fieldset>
</form>
```

【运行程序】那么，这个表单域在浏览器中的效果如图 12.13 所示。

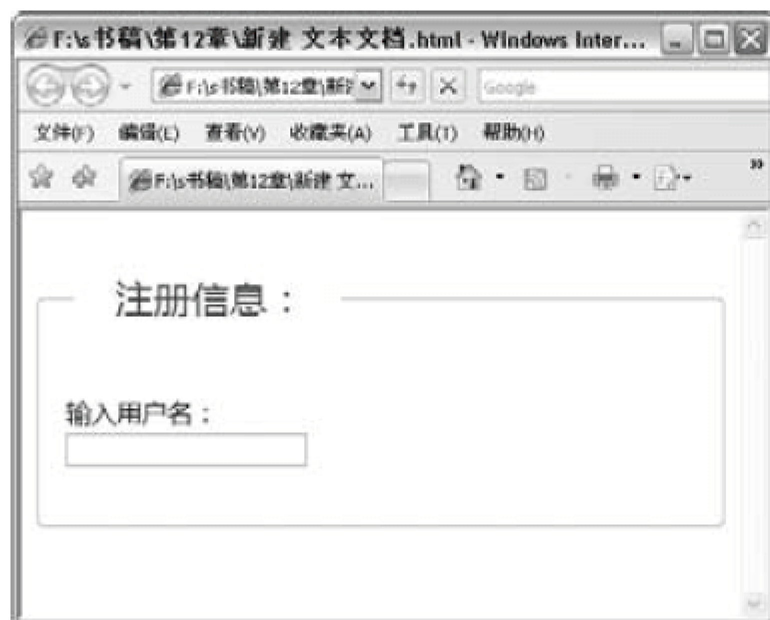


图 12.13　表单域

12.6　案例：制作一个精美的由表单构成的页面

在前面的例子中，读者应该反复感受到了表单也可以通过 CSS 样式表来进行修饰这样一个信息。是的，设计者可以通过样式表来修改表单中文本的样式，像框模型那样修饰表单的布局。在本例中，将使用更多的样式表来表现一个综合的页面，来配合页面中的表单，如程序 12.9 页面使用样式表来修饰的表单。

【本节示例参考：资料光盘\第 12 章\12-9　使用样式表修饰表单的页面.html】

【实例 12-9】使用样式表修饰表单的页面，其源码展示如下：

程序 12.9　使用样式表修饰表单的页面.html

```
001  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
002  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
003  <html xmlns="http://www.w3.org/1999/xhtml">
004    <head>
005      <title>使用样式表修饰表单的页面</title/>
006     <style>
007       body {background-color:white;
008             }
009       fieldset {border:2px dashed red;
010                 padding:10px;
011                 margin-top:20px;
012                 margin-bottom:20px;
013                  }
014       legend {font-family:微软雅黑;
015                font-size: 90%;
016                letter-spacing: -1px;
017                font-weight: bold;
018                line-height: 1.1;
019                color:#fff;
020                background: orange;
021                border: 1px solid #333;
022                padding: 2px 6px;
023               }
024       h1 {font-family:微软雅黑;
025           font-size: 175%;
026           letter-spacing: -1px;
027           font-weight: normal;
028           line-height: 1.1;
029           color:#333;
030           }
031       label {width:140px;
032              height:32px;
033              margin-top:3px;
034              margin-right:2px;
035              padding-top:11px;
036              padding-left:6px;
037              background-color:maroon;
038              float:left;
039              display: block;
040              font-family:幼圆;
041              font-size: 115%;
042              letter-spacing: -1px;
043              font-weight: normal;
044              line-height: 1.1;
045              color:yellow;
046              }
047      .form {margin:0;
```

设置表单域集合的样式表（009～013行）

设置表单域集合标题的样式表（014～023行）

设置 lable 对象的样式表，通过不同的空距、边框、边距的样式，来制作 lable 对象的样式（031～046行）

```
048              padding:0;
049              }
050        #container {width:750px;                //#container 设置页面主体在窗口中的位置
051                        margin:auto;
052                     padding:10px;
053               }
054        #top {width:680px;                      //#top 用来确定页面标题在页面中的位置
055               height:50px;
056              }
057        #leftSide {  width:530px;               //定义了页面中整个表单的位置
058                      padding-top:30px;
059                      float:left;
060                      }
061        .clear {clear:both;
062                      }
063        .holder {background-color:#fff;
064                     }
065
```

以上这部分样式表实现了两部分功能，第 9～49 行讲述了如何定义表单域集合的样式，以及在这个表单域页面中定义整个页面的布局。



说明：这里采用 leftSide 来命名表单的样式表，是为了说明如果设计者希望在表单右侧放入其他内容，可以通过定义一个 rightSide 层放置其他内容。事实上，在很多页面中都是这样做的。

```
066        .div_texbox {width:347px;                    //通过样式表制作表单域外部的样式
067                         float:right;
068                         height:35px;
069                         margin-top:3px;
070                         padding-top:5px;
071                         padding-bottom:3px;
072                         padding-left:5px;
073                          background-color:#E6E6E6;
074                          }
075        .textbox {background-color:#FFFFFF;          //通过样式表制作表单域内部的样式
076                   background-repeat: no-repeat;
077                   background-position:left;
078                   width:285px;
079                   font:normal 18px  微软雅黑;
080                   color: black;
081                   padding:3px 5px 3px 19px;          //通过设置不同大小的空距来制作阴影效果
082                     }
083        textbox:focus, .textbox:hover {background-color:orange;
084                                            }        //通过伪类来确定鼠标划过表单域的样式
```

```
085        .username {background-repeat: no-repeat;
086                        background-position:left;
087                        background-color:#FFFFFF;
088                        width:285px;
089                        font:normal 18px 微软雅黑;
090                        color: black;
091                        padding:3px 5px 3px 19px;
092                       }
093        .username:focus, .username:hover {background-color:orange;
094                                            }
095        .password {background-repeat: no-repeat;    //设置密码填写时的样式
096                       background-position:left;
097                       background-color:#FFFFFF;
098                       width:285px;
099                       font:normal 18px 微软雅黑;
100                       color: black;
101                       padding:3px 5px 3px 19px;
102                       }
103        .password:focus, .password:hover {background-color:orange;
104                                            }
```

这部分代码中的样式表定义了表单域是如何通过样式表来改变它的外表的，以及当鼠标划过表单域时将显示出怎样的效果。

```
105         .button_div
106                       {width:287px;                    //设置按钮的样式
107                        float:right;
108                        background-color:#fff;
109                        border:1px solid #ccc;
110                        text-align:right;
111                        height:35px;
112                        margin-top:3px;
113                        padding:5px 32px 3px;
114                         }
115        .buttons {background: #e3e3db;
116                   font-size:12px;
117                   color: #989070;
118                   padding: 6px 14px;
119                   border-width: 2px;
120                   border-style: solid;
121                   border-color: #fff #d8d8d0 #d8d8d0 #fff;
122                   text-decoration: none;
123                   text-transform:uppercase;
124                   font-weight:bold;
125                    }
126       </style>
```

```
127    </head>
128
```

代码的第 105～128 行定义了样式表中按钮的样式。以下部分的代码是页面的结构标签：

```
129    <body>
130      <div id="container">
131          <div id="top">
132              <h1>请填写这张表格：</h1>
133          </div>
134            <div id="leftSide">
135              <fieldset>
136                <legend>注册信息：</legend>
137                <form action="login.php" method="POST" class="form">
138                <label for="username">注册名：</label>
139                    <div class="div_texbox">
140                      <input name="username" type="text" class="username" id="username"
141  value="depp" />
142                  </div>
143                <label for="password">密码：</label>
144                <div class="div_texbox">
145                    <input name="password" type="password" class="password"
146  id="password" value="password" />
147                </div>
148                <div class="button_div">
149                    <input name="Submit" type="button" value="确  定" class="buttons" />
150                </div>
151              </form>
152      <div class="clear"></div>
153          </fieldset>
154          <hr size="1">
155          <fieldset>
156      <legend>个人信息：</legend>
157            <form action="..." method="POST" class="form">
158       <label for="name">姓名：</label>
159      <div class="div_texbox">
160      <input name="name" type="text" class="textbox" id="name" value="Depp" />
161       </div>
162       <label for="address">地址：</label>
163       <div class="div_texbox">
164      <input name="address" type="text" class="textbox" id="address" value="G 1128" />
165       </div>
166       <label for="city">所属区：</label>
167       <div class="div_texbox">
168      <input name="city" type="text" class="textbox" id="city" value="浦东新区" />
169       </div>
```

```
170          <label for="country">城市：</label>
171          <div class="div_texbox">
172         <input name="country" type="text" class="textbox" id="country" value="上海" />
173          </div>
174          <div class="button_div">
175          <input name="Submit" type="button" value="确  定" class="buttons" />
176          </div>
177                  </form>
178          </fieldset>
179        </div>
180      </div>
181    </body>
```

【**运行程序**】浏览该页面，结果如图 12.14 所示。



图 12.14　使用样式表修饰表单的页面

【**深入学习**】这个页面中使用了 20 个样式表，如果一步一步地去分析，它没有看上去那么可怕。body、fieldset、legend、h1 和 label 样式表，基于框模型的结构，定义了各自对象的字体的样式。.form、#container、#top 和#leftside 这 4 个样式表规划了页面的整体布局。#container 样式表定位了整个页面的大小，而#top 针对于页面的标题部分，#leftside 样式表则定义了表单部分的位置。

.div_textbox 定义了表单中文本域的样式，在例子中可以视为表单填写文本部分的外框。而.textbox 样式表定义了这个对象的内框。由于对 padding 属性设置了不同的值，而制作出了内嵌的阴影效果。类似的作用还有.button_div 和.buttons 的配合使用，它们配合的作用也是制作出按钮的内外边框的样式，来制造出按钮的立体效果。



注意：.textbox 和.buttons 样式表在 HTML 源码中的位置。

代码中第 93 和第 94 行，以及第 103 和 104 行，分别使用了伪类来定义当鼠标放在文本框中和当鼠标滑过文本框时的状态。

12.7　小　　结

本章介绍了表单的作用以及表单的表现形式，主要的知识点有：

- ❑ 表单的工作原理。
- ❑ 如何创建表单。
- ❑ 表单主要的样式以及它们的作用，通过这些表单，可以将使用者的信息递交给数据管理者。根据递交数据的样式不同，来决定使用怎样的表单。
- ❑ 了解表单中表单域的意义，它是用来给用户输入信息的区域。

在第 13 章中，将介绍 JavaScript 的入门知识，即 JavaScript 的基本语法。这是令你制作页面的水平真正走向高手的路径。

第 13 章　在网页中加入神奇的效果

在学习完 HTML 和 CSS 之后，设计者可以制作出一些精美的网页。但优秀的网站，绝不仅仅只是要求页面美观，或者说，无法互动的网页始终无法满足所有浏览用户的要求。为了让网页能有动态的变化，使得用户可以与网页进行交互，IE 提供了 Dynamic HTML 技术，简称为 DHTML。DHTML 主要由 3 个部分组成，分别为 HTML 网页标记、Script 语言与 CSS。最常用的脚本语言是 JavaScript。对于追求更高页面效果的设计者来说，JavaScript 是其在网页设计世界中立足的“利剑”。本章主要的知识点如下。

- 脚本语言的概念、常用的两种脚本语言介绍。
- 区别 JavaScript 和 Java。
- JavaScript 基本语法，包括标识符、基本数据类型、运算符、表达式、流程控制，以及函数。
- 通过一个案例来展示 JavaScript 基本语法的用法。

13.1　什么是脚本语言

脚本语言是一种解释性语言，不需要编译，一般用来编写嵌入在网页中的程序，由浏览器来负责解释执行。浏览器一般都由相对应的脚本引擎来解释执行，所以，支持脚本程序的浏览器需要集成用于解释脚本程序的模块。事实上，对于这些并没有听起来那么陌生。参照第 12 章中的程序 12.1，在 HTML 网页中，脚本程序代码是放在<script>标签中的，浏览器正是通过该标签来识别脚本程序的。

脚本语言一般以文本的形式存在，而不像 C、C++那样可以编译成二进制代码，也不用像 Java 那样解释生成.class 文件。通常在大多数网页设计中，开发者经常使用的脚本语言有 VBScript（Visual Basic Script）、JavaScript 和 ActionScript 等。目前最常用的两种脚本语言是 VBScript 和 JavaScript。

13.1.1　初识 VBScript

VBScript 是基于微软 Microsoft 公司的 Visual Basic。其广泛应用于网页和 ASP 程序制作，因为其与 Visual Basic 的紧密联系，所以对于拥有 Visual Basic 开发经验的程序员来说，很容易入门。现在通过一个简单的示例来体验它的效果，如程序 13.1 所示为一个简单的 VBScript 示例。

【本节示例参考：资料光盘\第 13 章\13-1　VBScript 简单示例.html】

【实例 13-1】VBScript 简单示例，其源码展示如下：

程序 13.1　VBScript简单示例.html

```
01	<html>
02	<head>
```

```
03          <title>VBScript 简单使用</title>
04          <script language="VBScript">
05          <!--
06            Sub Button1_OnClick
07                MsgBox "Hello！这是 VBScript 代码产生的效果"
08            End Sub
09          -->
10          </script>
11      </head>
12      <body>
13          <h3>VBScript 简单使用</h3><HR>
14          <form>
15              <input name="Button1" type="BUTTON" value="显示 VBScript">
16          </form>
17      </body>
18      </html>
```



注意：浏览器是通过 language="VBScript"来知道执行的脚本语言是 VBScript 的。

将上面的代码粘贴到 vbscript.html 文件的源代码中，单击浏览器中的“刷新”按钮，运行结果如图 13.1 所示。之后再次单击“显示 VBScript 效果”按钮，显示结果如图 13.2 所示。



图 13.1　一个 VBScript 编写的简单页面



图 13.2　单击“显示 VBScript 效果”按钮的显示结果

在单击“显示 VBScript 效果”按钮后，会出现一个对话框，这个对话框就是用 VBScript 实现的，即程序 13.1 中的第 4～10 行的代码。

13.1.2　学习 JavaScript 的起步

JavaScript 是 Netscape 公司借鉴 Sun 公司的 Java 的相关概念，将其自身的 Livescript 进行重新设计之后推出的。因此，JavaScript 的很多语法都与 C++、Java 的风格非常相似。学习过这些编程语言将有助于学习 JavaScript。与 C++、Java 等编程语言类似，在 JavaScript 中也包含类、对象、变量、函数等，而且使用的流程控制也基本相似。不同之处在于 JavaScript 的语法规则更松散，不像编程语言那么复杂。

例如，对于变量的定义，在 JavaScript 中只需通过 var 定义即可，而不必像编程语言中定义其为 int 或 char。此外，由于 JavaScript 代码不会被编译为二进制代码文件，只是作为一种网页文件（在本书中指 HTML 文件）的一部分由浏览器解释执行。因此，修改起来也要比编程语言在集成平台中方便。所以 JavaScript 是一种简单易学的语言，但又是一种具有强大特效功能的语言。同样，现在通过一个简单的示例来初步了解一下 JavaScript 的魅力，如程序 13.2 展示的是和程序 13.1 同样效果的页面。不同的是这里使用的脚本语言是 JavaScript 语言。

【本节示例参考：资料光盘\第 13 章\13-2　JavaScript 简单示例.html】

【实例 13-2】JavaScript 简单示例，其源码展示如下：

程序 13.2　JavaScript简单示例.html

```
01          <html>
02            <head>
03                <title>JavaScript 简单使用</title>
04              <script language="JavaScript">
05                function button1()
06              {
07               alert("Hello!这是 JavaScript 的显示效果");
08              }
09              </script>
10            </head>
11            <body>
12              <center>
13              <h3>JavaScript 简单使用</h3><HR>
14              </center>
15              <form>
16                 <input name="Button1" type="BUTTON" value="显示 JavaScript 效果"
17          onclick="button1()">
18              </form>
19            </body>
20          </html>
```



注意：浏览器是通过 language="JavaScript"来知道执行的脚本语言是 JavaScript。

【运行程序】将上面的代码直接粘贴在 javascript.html 文件的源代码中，单击“刷新”按钮，运行结果如图 13.3 所示。再次单击“显示 JavaScript 效果”按钮，运行结果如图 13.4 所示。



说明：读者可能暂时不理解 JavaScript 的部分代码，在后续的章节中将会详细讲解如何使用 JavaScript 设计漂亮的网页。



图 13.3　一个 JavaScript 编写的简单页面

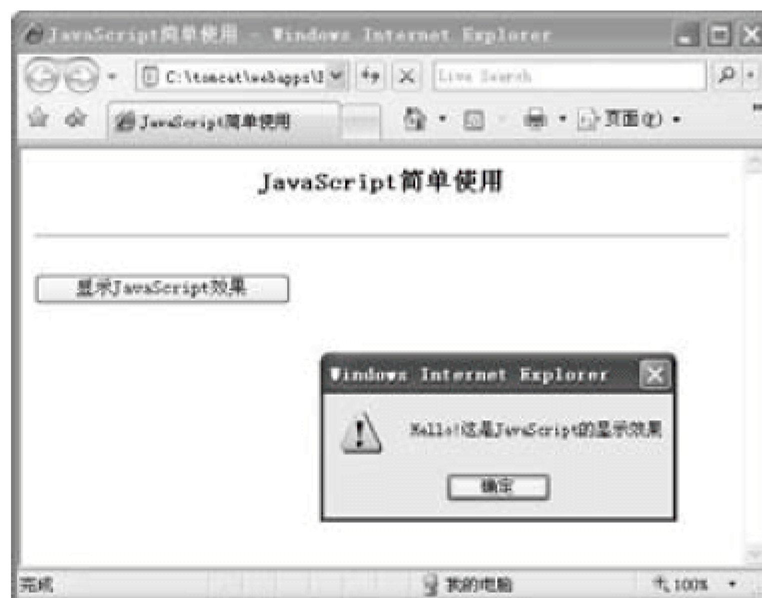


图 13.4　单击“显示 JavaScript 效果”按钮的显示结果

【深入学习】可以看到，用 JavaScript 同样实现了上面 VBScript 例子的功能。单击“显示 JavaScript 效果”按钮之后，同样出现一个对话框，而此处是使用程序 13.2 中第 4～9 行的 JavaScript 代码实现的。另外，在使用 button 按钮时需要通过 onclick 指明调用的函数，如程序 13.2 中的第 17 行代码所示。

正是由于 JavaScript 与 C++、Java 等编程语言的相似性，且易阅读、易掌握，所以大部分网页设计者偏爱使用 JavaScript。对于初学者来说，学习一种编程技术时跟随趋势是一条很不错的捷径，往往可以事半功倍。因为 JavaScript 的发展已比较成熟，供初学者参考的范例和可借鉴的经验也较多，这样就可以大大减少在学习中可能出现的问题。本书主要介绍的是 JavaScript。

13.2　JavaScript 和 Java 的区别

在 13.1 节中提到，JavaScript 与目前盛行的编程语言 Java 比较相似，然而对于学习者来说，需要明确 JavaScript 与 Java 是不一样的，有很多初学者不明白这两者的区别。事实上，它们并不像字面看上去那样联系紧密，JavaScript 并不是 Java 的子集，而是有很大的区别，不理解这点可能会掉入混为一谈的误区。在日常工作、学习中不乏将两者混淆的例子，经常有编程者把 Java 中出现的问题提交到 JavaScript 的学习交流社区，或者将 JavaScript 的难题求助于 Java 学习社区。

JavaScript 和 Java 是分别来自两个不同公司的产品，这很显然地决定了两者有很大的差异。Java 是 SUN 公司推出的一种面向对象的程序设计语言，非常适合于 Internet 应用程序的开发。而 JavaScript 是 Netscape 公司为了扩展 Netscape Navigator 功能而开发的产品，是一种基于对象和事件驱动的解释性语言，通常嵌入在 Web 页面中。

JavaScript 借鉴了 Java 的相关概念，所以两者在语法上有很多相似之处，只是不完全相同。例如，其各自所使用的变量是不一样的。JavaScript 是弱类型变量声明，即没有非常严格的类型声明要求，在定义时只需要使用 var 定义即可。在运行时由解释器检查其数据类型（即动态联编），而 Java 采用强类型变量检查，所有变量在使用前必须声明数据类型（即静态联编）。

此外，两者在嵌入方式上也不一样，Java 通过<applet>标签来标识，是一种与 HTML 无关的格式。其代码以字节代码的形式保存在独立的文档中，必须通过 HTML 引用装载。而 JavaScript 是以文本的形式存在，可以直接放在<script>标签之间。

当明白了 JavaScript 与 Java 的区别之后，对于有 Java 开发经验的读者来说，可以避免将 Java 中的惯性带入到 JavaScript 中。而对于页面开发的初学者来说，也不用惧怕所谓 Java 名头的来势汹汹。

13.3　JavaScript 的基本语法

前面介绍了什么是 JavaScript，以及 JavaScript 的特点，并且通过一个简单的 JavaScript 示例实现了单击按钮弹出对话框的功能。对比之前讲过的 HTML，设计者可以初步体验到 JavaScript 与用户互动的特效功能。然而这个示例只是冰山一角，还不足以完全展现 JavaScript 的魅力。从本节开始将详细地学习 JavaScript，像其他编程语言一样，需要先从基本的语法开始。

13.3.1　JavaScript 中的标识符和保留关键字

也许有 C、C++或者 Java 编程经验的人对标识符这个概念已经不陌生了，而 JavaScript 中的标识符与其他编程语言中的概念基本一样，是指 JavaScript 中定义的符号，必须以字母、下划线（_）或美元符号（$）开始。其他字符可以是字母、数字、下划线或美元符号，如变量名、函数名等。但是，标识符不能是 JavaScript 中的保留关键字且不能包含空格。

例如，下面都是定义“电话号码”的合法标识符。

```
telephoneNum
telephone_Num
_telephoneNum
$telephoneNum
tl
```

JavaScript 的标识符对大小写敏感，telephoneNum 和 telephonenum 是两个不同的标识符，这是在编写代码过程中很容易出现的疏忽。此外，在实际定义标识符时可根据大多数人的习惯来定义，一般用第一种定义方法，如 telephoneNum。利用名字组合可以望文生义，便于理解，如果定义成 tl，则别人就很难理解了。名字组合时第一个单词首字符小写，后边的单词首字母均大写，这样做既便于阅读，又便于输入。注意，下面都是非法标识符：

```
This                          this 是 JavaScript 中的保留关键字
2008_Olympic                  标识符不能以数字开头
2008.08                       标识符中不能含有点（.）
One World                     标识符中不能含有空格
```

使用者在编写时为了避免使用保留关键字来定义标识符，可以参考表 13.1 中总结的 JavaScript 保留关键字。



说明：除了表 13.1 中的保留关键字，在定义标识符时还要避免那些已经用作 JavaScript 的内部对象或函数的名称的词，如 string 或 alert 等单词。当在运行时提示标识符错误，如果不在上边介绍的非法字符之列，不妨检查一下是否用了已定义好的内部对象或函数的名称。

表 13.1　JavaScript保留关键字

abstract	boolean	Break
byte	case	Catch
char	class	const
continue	default	Do
double	else	extends
false	finally	Float
for	function	Goto
in	instanceof	Int
if	implements	import
interface	long	Native
new	null	package
private	protected	Public
return	short	Static
super	switch	synchronized
this	throw	throws
transient	true	Try
var	void	While
with		

13.3.2　JavaScript 语法的特殊规则

JavaScript 对大小写非常敏感，在程序中定义 xman 和 Xman 是不同的，这是两个变量。前面已经提过，在此处不厌其烦地再提是因为这是一个初学者犯错率很高的知识点。HTML 是对大小写不敏感的，xman 和 Xman 是一样的。将 JavaScript 嵌入在 HTML 中很容易混淆，要保持高度的警惕性。

一般情况下，JavaScript 每条执行语句的后面都要以分号（;）来结束。但是当 JavaScript 的代码作为属性值时，最后一句后面的分号可以省略，例如：

```
action="javascript:checkDay()"            //省略了最后的分号
```

注意英文标点符号和中文标点符号的区别，在 JavaScript 中用的都是英文标点符号，不只是初学者，就是身经百战的开发者，有时也会因为马虎将“,”写成“，”，将“;”写成“；”。这也是在调试程序时应该注重查找的地方。



注意：在 JavaScript 中，“//”表示注释一行代码，注释多行代码时使用“/*”开头，以“*/”结尾。另外，“/* */”中可以嵌套“//”，但不能嵌套“/* */”，因为第一个“/*”会与其后面第一次出现的“*/”配对，例如：

```
<script language="javascript">
/*                                          //第一个注释符
   /*下面的代码会弹出一个警告框*/              //第二个注释符
   警告框中的内容是“好运 2008！”
*/
alert("好运 2008");
</script>
```

上面代码中第一次出现的“*/”会与第一个“/*”配对，导致警告框中的内容 “好运 2008！”没有被注释掉，所以 JavaScript 无法解释这个字符串，打开页面时就会出现错误。

13.4　JavaScript 的数据类型

JavaScript 数据类型包括基本数据类型和内置数据类型。基本数据类型一般包括整型、实型、字符串型、布尔型和空值 5 种。基本数据类型定义的数据可以是常量，也可以是变量。JavaScript 的常量又称为字面常量，其值不能随便改变。变量是程序向系统申请的内存单元，用来存储各种类型的数据。

13.4.1　常量

与基本数据类型相对应，常量一般分为 5 种，分别是整型常量、实型常量、布尔型常量、字符型常量和空值。在编写 JavaScript 程序时，常量数据类型是应用广泛、常见的一种类型。常量通常类似于可以看得到、能想象出来的元素。

- 整型常量：一般来说，整型常量可以采用十进制、八进制和十六进制来表示。十进制的首位不能是数字 0，如 2008。八进制以 0 为首位，如 0351。十六进制以 0x 或 oX 开头，如 0x86 或 oX86。这 3 种进制是可以相互转化的。
- 实型常量：可以采用整数部分加小数部分的形式来表示的值，也可以采用科学计数法来表示，如实数 1000.00 用科学计算法表示是 1E3 或 1e3。
- 布尔型常量：有两个值 true 和 false。通常在流程控制中作为判断条件。
- 字符型常量：是用单引号（'）或双引号（"）引起来的零个、一个或几个字符，如"One World One Dream"、"a"、""。""表示一个空字符串。



注意：同 Java 语言一样，JavaScript 中以反斜杠（\）作为转义字符来表示一些特殊字符，如\\、\n 等。\\表示斜杠，\n 表示换行。

- 空值：即表示什么也没有，当引用没有定义的变量时，就返回一个 null 值。

13.4.2　变量

变量并非是指一个变换不定的元素，变量相当于设置好一个位置或一个符号，所以使用者可以将不同的元素定义为这个位置或符号。这就是变量的含义。JavaScript 中采用弱类型的变量形式，即声明一个变量时不必指明其为整型还是字符型，而是使用关键字 var 声明，例如：

```
var telephoneNum;
```

这条语句定义了一个变量，即申请了内存。但还没有值，可以在使用时为其赋值，如将一个数值赋给这个变量。

```
telephoneNum=62286688;
```

也可以在变量声明时就为其赋值，例如：

```
var telephoneNum=62286688;
```

上面的定义方法与其他编程语言中的定义方法是一样的。JavaScript 的特殊之处在于可以不事先声明变量而直接使用。浏览器在解释执行到该语句时，会自动产生一个相应类型的变量，例如：

```
school="BUPT";
```

浏览器在解释执行上面的语句时就会自动产生一个字符串型变量。比较好的方法是偏向于事先声明变量，这样做的好处是能及时发现代码中的错误。因为 JavaScript 是采用动态编译的，而动态编译不易发现代码中的错误，特别是变量命名的方面。

13.4.3　数据类型转换

在数据类型的使用过程中，经常会出现返回值并不是要求的数据类型的情况，这就需要在不同数据类型之间进行转换。JavaScript 中数据类型的转换方法有两种：一是将整个值从一种类型转换为另一种数据类型；二是从一个值提取另一种类型的值，并完成转换工作。



注意：13.4.4 小节要讲的表达式运算中也需要统一数据类型。

1．一种数据类型转换为另一种数据类型

这种情况有 3 种转换方法，分别是 String()、Number()和 Boolean()方法。

- String()方法：表示将任意一种数据类型转换为字符型。例如：

```
String(2008);
```

其转换结果为"2008"。

- Number()方法：将任意一种数据类型转换为数值型。例如：

```
Number("2008");
```

其转换结果为 2008。

- Boolean()方法：将任意一种数据类型转换为布尔型。例如：

```
Boolean("aaa");
```

其转换结果为 true。

2．从一个值提取另一种类型的值，并完成转换工作

这种情况也有 3 种转换方法，分别是 parseInt()、parseFloat()和 eval()。

- parseInt()：表示提取字符串中的整数。例如：

```
parseInt("2008Olympic");
```

其转换结果为 2008。

- parseFloat()：表示提取字符串中的浮点数。例如：

```
parseFloat("2008.08Olympic");
```

其转换结果为 2008.08。

- eval()：表示执行用字符串表示的一段 JavaScript 代码。例如：

```
nextOlympic=eval("2008+4");
```

其转换结果为 nextOlympic=2012。

13.4.4　运算符

JavaScript 是用来处理对象运算的符号，是具有全范围的运算符。按照处理对象的数目分为单元运算符、二元运算符和三元运算符。更常见的分类方法是按照功能分为算术运算符、赋值运算符、比较运算符、逻辑运算符和位运算符。

1．算术运算符

JavaScript 中常用的算术运算符如表 13.2 所示。

表 13.2　算术运算符

运　算　符	运算符说明	示　　例
+	加法运算符或正值运算符	y+2008，+2
−	减法运算符或负值运算符	100−8，−9
*	乘法运算符	3*5
/	除法运算符	12/4
%	求模运算符	5%3



注意：当表达式中至少有一个字符串时，“+”表示多个字符串的连接，如"Olympic"+2008 的结果是"Olympic2008"。

2．赋值运算符

赋值运算符是将其右边的一个值或者表达式的值赋给其左边的变量。常用的赋值运算符如表 13.3 所示。

表 13.3　常用的赋值运算符

运　算　符	运算符说明	示　　例
++	将变量值加1后再将结果赋给这个变量	++1，1++
−−	将变量值减1后再将结果赋给这个变量	−−1，1−−

++有两种用法：++y 和 y++。前者是变量先将自己加 1，再参与其他运算，而后者是变量在参与其他运算后，再将自己加 1。−−与++用法一样。

3．比较运算符

比较运算符用在逻辑语句中，比较两边的操作数，返回一个布尔值，结果为真时返回 true，结果为假时返回 false。常用的比较运算符如表 13.4 所示。

表 13.4　常用的比较运算符

运　算　符	运算符说明	示　　例
>	当左边操作数大于右边操作数时返回true，否则返回false	4>3返回true，5>6返回false
<	当左边操作数小于右边操作数时返回true，否则返回false	4<5返回true，5<1返回false
>=	当左边操作数大于等于右边操作数时返回true，否则返回false	4>=4返回true，4>=5返回false
<=	当左边操作数小于等于右边操作数时返回true，否则返回false	4<=4返回true，4<=1返回false
==	当左边操作数等于右边操作数时返回true，否则返回false	5==5返回true，5==6返回false
!=	当左边操作数不等于右边操作数时返回true，否则返回false	5!=8返回true，5!=5返回false



注意：区分比较运算符“= =”与赋值运算符“=”。

4. 逻辑运算符

逻辑运算符用于测定变量和值之间的逻辑，采用布尔值 true 或 false 作为操作数，其返回值也是逻辑值。常用的逻辑运算符如表 13.5 所示，假定 x=8 且 y=5。

表 13.5　常用的逻辑运算符

运　算　符	运算符说明	示　　例
&&	当左右两边操作数均为true时返回true，否则返回false	(x<10)&&(y>4)返回true
||	当左右两边操作数至少有一个为true时返回true，否则返回false	(x<=10)&&(y<2)返回true
!	当操作数为true时返回false，否则返回true	!(x<10)返回false

5. 位运算符

位运算符包括位逻辑运算符和位移动运算符。位逻辑运算符有 3 种，即&、|、^。位移动运算符有 3 种，即>>、<<、>>>。位运算符在编写 JavaScript 程序时不太常用，此处不做详细介绍，需要时可参考相关手册。

13.4.5　表达式

在学习完变量和运算符之后，表达式的概念就很容易理解了，其实在前面已经多次见到简单的表达式。表达式是变量、常量及运算符的集合。一般分为算术表达式、字符串表达式、赋值表达式，以及关系表达式等。

表达式的种类是多种多样的，当表达式中包含多个运算符时，运算符的优先级就显得非常重要了。如表 13.6 所示为按优先级从高到低列出的 JavaScript 运算符，具有相同优先级的运算符按从左至右的顺序求值。

表 13.6　JavaScript运算符优先级

优　先　级	运　算　符
1	。[] ()
2	++ -- - ~ !
3	* / %
4	+ - +（字符串链接）
5	<< >> >>>

续表

优　先　级	运　算　符
6	< <= > >=
7	&（按位与）^（按位异或）|（按位或）
8	&&（逻辑与）||（逻辑或）
9	?:（条件运算符）
10	=（赋值运算符）
11	,（多重求值）

来看一个表达式，例如：

```
y=2008*(55+6+23);
```

在该表达式中有 5 个运算符，即=、*、()、+、+。按照表 13.6 的优先级规则从高到低应为()、+、+、*、=。

13.5 流 程 控 制

程序并不都是按部就班地执行，这就需要控制结构来大显神威。日常生活中经常会根据不同的情况做不同的决定。例如，下雨就呆在家里看奥运直播，或者不下雨就跟朋友出去逛街，不同条件下触发不同的流程结果，再如在工厂中控制程序控制机器不停地执行组装汽车零件。这些情况的处理就离不开流程控制了。JavaScript 常用的程序流程有 3 种结构，即顺序结构、选择结构和循环结构。

13.5.1 顺序结构

顺序结构是最基本的控制结构，任何一个程序都离不开，是程序按照自上而下的顺序逐行执行的。例如：

```
var x=2008;
alert(x);              //弹出一个对话框显示变量 x 的值
```

此例就是典型的顺序结构，浏览器先初始化变量 x，并为其赋值 2008。然后再执行下一行，弹出一个对话框来显示 x 的值。

13.5.2 选择结构

重用的选择结构有 if 结构、if…else 结构和 switch 结构。选择结构即是像“如果不是…那么将…”这样形式的结构。

1. if 结构

if 结构的格式写法是：

```
if(条件语句)
   {
       语句
    }
```

当条件语句的返回结果是 true 时则程序执行大括号中的语句，然后顺序执行后面的其他程序。如果条件语句返回 false，则程序不执行大括号中的语句，而直接执行后面的其他程序。例如：

```
var y=10;
  if(y>9)
   {
     alert(y);
     }
```

上面的程序中，当条件语句 y 值大于 9 时，语句返回值为 true，那么这个程序将执行 alert(y)。



注意：虽然上面的 if 从句只有一条语句，但建议不要省略大括号对。这样做易读、易维护。程序员平时应养成良好的习惯。

2．if…else 结构

if…else 的格式是：

```
if(条件语句)
{
  语句 1
}
else
{
  语句 2
}
```

这种结构是当条件语句为 true 时，程序执行语句 1；而条件语句为 false 时，程序执行语句 2。例如：

```
var y=2008;
if(y==2008)
{
alert("奥运年");
}
else
{
alert("年年有余");
}
```

如果 y 的值等于 2008，则弹出对话框显示“奥运年”；否则将弹出对话框“年年有余”。另外，if 语句可以嵌套使用。

3．switch 结构

对于条件语句拥有多个值时，使用 switch 语句就会显得得心应手。其语句的格式是：

```
switch(表达式)
{
case 取值 1:
    语句 1
```

```
        break;
case  取值 2:
        语句 2
        break;
case  取值 3:
        语句 3
        break;
…
case  取值 n:
        语句 n
        break;
default:
语句 n+1
break;
}
```

程序根据 switch 表达式的结果来与 case 后面的值进行匹配。如果与 n 匹配，则执行语句 n，直到碰到 break 语句为止。default 语句是可选的，只有上面的值都不匹配时，才执行语句 n+1。例如：

```
switch(month)
{
   case February:
          alert("28 天");
          break;
   case April:
   case June:
   case September:
   case November:
          alert("30 天");
          break;
   default:
          alert("31 天");
          break;
}
```

此例中如果月份（month）等于 February，则弹出对话框显示“28 天”；如果是 April、June、September 或者 November 中的任意一个，则弹出对话框显示“30 天”；其他月份弹出对话框显示“31 天”。



注意：month 的值只能是整型或者字符串，不能是实型。

为了完全理解上面的代码，此处需要说明一下 break 语句的用法。break 语句可以终止循环体中的执行语句和 switch 语句。一般来说，循环条件为 false 时循环才结束。如果提前中断循环，可以在循环体语句中添加 break 语句。除了 break，还有 continue 语句来中断循环，不过 continue 只是跳过本次循环要执行的剩余语句，继续执行下一次循环。要注意 break 和 continue 的区别。

13.5.3　循环结构

循环结构一般有 while 结构、do…while 结构和 for 结构。它们都是表示一个动作完成之后继续重复上一个动作。

1．while 结构

该结构的格式是：

```
while(条件表达式)
{
 语句
}
```

当条件表达式的返回值为 true 时，就执行大括号中的语句。当条件表达式的返回值为 false 时，就跳出循环，执行后面的语句。例如：

```
var i=0;
while(i<10)
{
   alert("我在循环中"+i);
   i++;
}
```

上述代码中，i 从 0 增加到 9，每次都弹出对话框，当 i=10 时，条件表达式 i<10 为 false，则跳出循环，执行 while 循环后面的代码。

2．do…while 结构

do…while 的格式是：

```
do
{
    语句
}while(条件表达式);
```

其结构与 while 结构的不同之处在于，do…while 是先执行大括号中的语句，再检查条件表达式的值，因此大括号中的代码至少要被执行一次。例如：

```
var i=0;
do
{
   alert("我在循环中"+i);
   i++;
} while(i<10)
```

此例在执行时会弹出 11 次对话框，可以与 while 结构中的例子做一下对比，便于理解两者的不同。

3．for 结构

最常用的循环是 for 循环，在学习其用法之后，可与其他循环结构对比，根据个人习惯选择使用哪种。for 循环语句的格式是：

```
for(初始值;循环条件;更新值)
{
  语句
}
```

通过 for 循环结构的用法，可以实现同 while 循环同样的作用。以下代码的效果和前面 while 例子是一样的。

```
for(var i=0;i<10;i++)
{
  alert("我在循环中"+i);
}
```



注意：for 关键字后面的小括号中是用两个“;”，而不是“,”。初学者比较容易忽略这个问题。在刚开始编写 JavaScript 代码时稍加注意即可。

如果修改 while 结构中的代码。例如：

```
var i=0;
while(i<10)
{
  alert("我在循环中"+i);
  break;
  i++;
}
```

此时因为在循环结构中加了 break 语句，循环只会执行一次，即弹出一次对话框之后循环中断，继续执行 for 循环后面的语句。再将代码改为：

```
var i=0;
while(i<10)                    //while 循环
{
  alert("我在循环中"+i);
  continue;                    //跳出本次循环
  i++;
}
```



注意：这是一个无效的错误循环。

此时又是另一种效果，执行程序时代码会弹出无数次对话框。事实上此程序已成为一个无限循环，即死循环。因为每次执行到 continue 语句时就跳出本次循环，i 的值没有改变，继续调入下次循环，这样的结果是 i 始终为 0，i<10 始终为 true。熟练掌握 break 和 continue 这两个语句对于以后编程很有用。

13.6　了 解 函 数

在编写程序时，经常有几处或者更多地方需要相同的功能。如果在每一处均写相同的代码，这将使得程序显得冗余。例如，在写射击类游戏程序时，最常用的一段代码就是发射子弹，在这样一个程序中，需要发射子弹代码的地方多则有几十处甚至上百处。如果在每次都写一段发射子弹的程序，这样显然需要重复编写大量的代码，费时费力达到相同的效果也无可非议。

但是当用户要求改变发射子弹的时间间隔时，就会觉察到这样做的缺点。设计者不得不一处接一处地修改，稍不注意疏漏一处，不得不重新检查，这就是恶性循环。而如果将发射子弹的代码独立作为一个函数，每次使用时直接调用即可，不仅节省了大量代码，而且在需要修改时只修改一处即可。所以函数可以令程序易编写、易读、易维护，一举三得。通常情况下，函数的语法格式如下：

```
function  函数名([参数 1][,参数 2]…[,参数 n])
{
    语句
    [return  表达式;]
}
```

其中，function 是定义函数的必需关键字。函数名的定义与前面讲的变量定义一样。小括号中的参数其实就是变量，各个变量之间用逗号（,）隔开，是为了接收调用程序传递进来的参数。当函数不需要接收任何参数时，小括号中什么也不写，但不能省略小括号。

如果调用程序需要返回一个结果，则在定义时要加 return 语句，返回结果。下面通过例子来说明如何定义函数和如何调用函数，如程序 13.3 所示为一个简单的函数调用示例。

【本节示例参考：资料光盘\第 13 章\13-3　简单的函数定义和调用示例.html】

【实例 13-3】简单的函数定义和调用示例，其源码展示如下：

程序 13.3　简单的函数定义和调用示例.html

```
01          <script language="JavaScript">
02            var msg="全局变量";
03            function maxValue(a,b)
04            {
05              var max=-10000;
06              if(a<b)
07                 {
08                   max=b;
09                 }
10              else
11                 {
12                   max=a;
13                 }
14              return max;
```

```
15                }
16              function checkVar()
17              {
18                 var msg="局部变量";
19                 alert(msg);
20              }
21
22              var max;
23                alert("max="+maxValue(23,48));
24                //checkVar();
25                alert(msg);
26          </script>
```

【运行程序】上述代码运行的结果如图 13.5 所示。在图 13.5 中单击“确定”按钮后如图 13.6 所示。

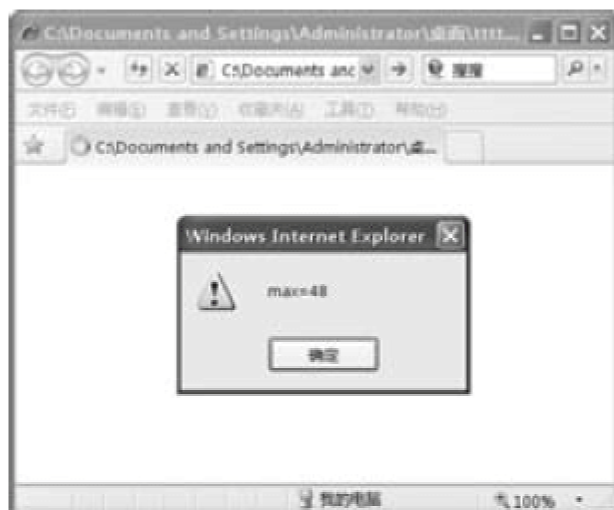


图 13.5　函数定义和调用演示一

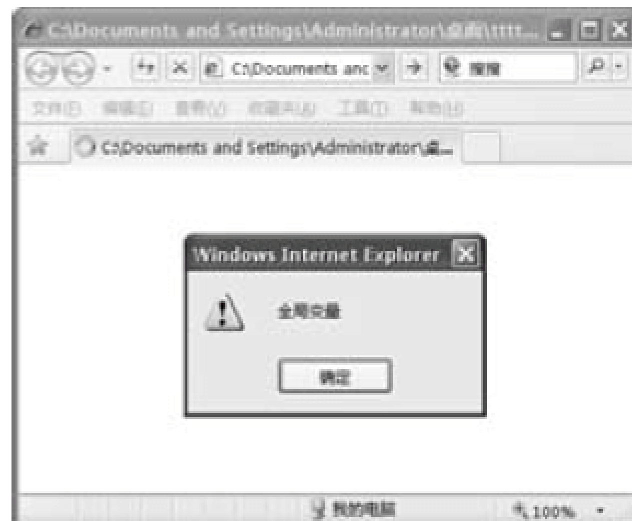


图 13.6　函数定义和调用演示二

【深入学习】在上面的代码中，定义了两个函数 maxValie()和 checkVar()，前者是一个有参数和返回结果的函数，通过接收调用函数传递的两个值，求出最大值。然后将结果返回给调用函数。如图 13.5 中的“48”正是函数 maxValue()的返回结果。而 checkVar()是一个没有参数、没有返回结果的函数，用来显示全局变量和局部变量的不同。

全局变量是在所有函数之外定义的变量，其作用域是<script>标签之间的所有代码。而局部变量是定义在函数之内的变量，其只在所在函数中有效。如图 13.6 所示，运行结果显示全局变量，而不是局部变量。当将程序 13.3 中第 24 行和第 25 行代码改为：

```
24      checkVar();
25      //alert(msg);
```

此时程序调用局部变量所在的函数，运行结果如图 13.7 所示。

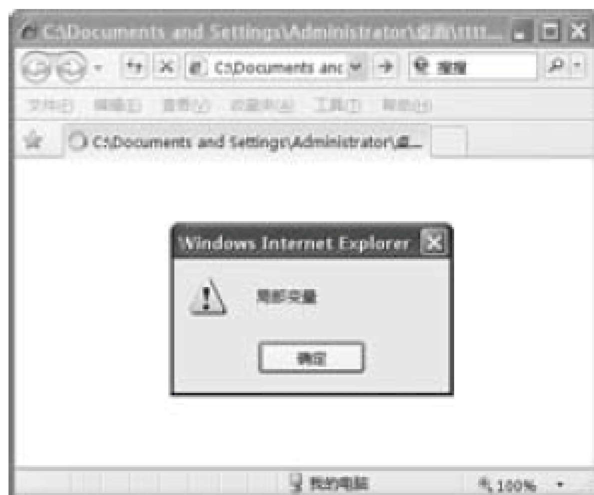


图 13.7　检查局部变量和全局变量的作用域

另外，由图 13.6 和图 13.7 对比可以知道，浏览器在按顺序解释执行嵌在网页中的脚本时，不会自动解释执行位于函数中的程序代码。只有当程序调用函数时，函数中的代码才会执行。



注意：准确区分“全局变量”和“局部变量”的概念，避免在编写程序时因为混淆全局变量和局部变量而导致程序运行结果不正确。

在编写 JavaScript 时并不是所有的函数都必须自己写，JavaScript 的开发者在设计时已经提供了一些系统函数供 JavaScript 程序员来使用。常用的 JavaScript 系统函数如表 13.7 所示。

表 13.7　常用的JavaScript系统函数

类　别	函　数	说　明	示　例
编码处理函数	encodeURI	返回一个URI字符串编码后的结果	encodeURI("http://www.sina.com/天气")的结果为 http://www.sina.com/%E5%A4%A9%E6%B0%94
	decodeURI	将已编码的URI字符串解码成原始的字符串返回	decodeURI("http://www.sina.com/%E5%A4%A9%E6%B0%94")的结果为http://www.sina.com/天气
数值处理函数	parseInt	将一个字符串指定的进制转换为一个整数	parseInt("2008",10);将字符串"2008"转换为十进制，结果为2008 parseInt("11",8);将字符串"10"转换为八进制，结果为9
	parseFloat	将一个字符串转换成对应的小数	parseFloat("6.6")+1的结果为7.6，注意区别 parseFloat("6.6"+1)的结果为6.61
	isNaN	检测前两个方法返回值是否为非数值型，如果是返回true，否则返回false	isNaN(parseInt("Olympic2008"))的结果为true
字符串编码处理函数	escape	返回一个字符串编码后的结果字符串，所有字符都用%xx编码，其中xx是表示该字符的Unicode编码的十六进制数	escape("Olym^_^pic")的结果为Olym%5E_%5Epic
	unescape	将用escape方法编码的结果字符串编码成原始字符串	unescape("Olym%5E_%5Epic")的结果为Olym^_^pic
	eval	将某个参数字符串作为一个JavaScript执行	eval("var a"+8+"="+8)相当于var a8=8

13.7　案例：一个使用基本语法的 JavaScript 例子

在日常生活中，对于上班时间比较灵活的单位来说，如何能够及时、准确地令公司员工了解自己的上班时间是一个棘手的问题，公司的管理人员不会一个挨一个电话通知，而难免总有员工会马虎弄错自己的班时。这样，可以求助于 Web，在公司的主页中设计一个员工上班时间查询系统，那么公司的员工就可以通过网络很方便地查询自己的工作日期。

要开发的员工上班时间查询系统的思路是可以在员工登录到查询系统页面后，填写当天是星期几，然后通过自己的员工号，来指定个人的上班时间查询，以及返回该员工是第几位访客。在这个例子中，程序要求员工号必须是 6 位，且每一位都是数字。

按照构思的需求，设计一个简单的原型系统，并不会包含所有实际需要的功能，如程序 13.4 展示的员工上班时间查询系统。

【本节示例参考：资料光盘\第 13 章\13-4　员工上班时间查询系统.html】

【实例 13-4】员工上班时间查询系统，其源码展示如下：

程序 13.4　员工上班时间查询系统.html

```
01  <html>
02   <head>
03   <title>员工查询系统</title>
04          </head>
05          <script language="javascript">
06                 var sum=0;
07                 function dosubmit(frm)          //检查员工号是否为 6 位，是否全是数字
08                 {
09                        if(frm.num.value.length!=6)
10                        {
11                               alert("员工号必须是 6 位");
12                               return false;          //如果不是 6 位，则返回错误
13                               }
14                               else
15                                      {
16                                             var num_value=frm.num.value;
17                                             for(var i=0;i<num_value.length;i++)
18                                             {
19                                                    if(num_value.charAt(i)<'0'||num_value.charAt(i)>'9')
20                                                    {
21                                                           alert("员工号只能是数字");
22                                                           return false;
23                                                           }
24                                                           }
25                                             }
26                                             return true;                    //alert("输入正确")
27                        }
28               /*返回星期几的上班时间，周一至周五返回"上班时间：9:00-17:30"，周六周日
29          返回"周末休息"*/
30               function checkDay()
31                 {
32                 switch(parseInt(form1.day.value))
33                 {
34                 case 1:
```

```
35                    case 2:
36                    case 3:
37                    case 4:
38                    case 5:
39                            alert("上班时间：9:00-17:30");
40                              break;
41
42                    default:
43                            alert("周末休息");
44                            break;
45                    }
46                      sum+=1;
47                      alert("您是第"+sum+"位访客");
48             }
49             </script>
50             <body>
51             <center>
52             <h1>员工查询系统</h1>
53             </center>
54             <form name=form1 action="javascript:checkDay()" method=post onsubmit="return
55           dosubmit(this)">
56                          星期：    <input type=text name=day><br>
57                       员工号：<input type=text name=num ><br>
58                  <input type=submit name=submit1 value="递交">
59             </form>
60             </body>
61             </html>
```



注意：本例旨在让读者体会 JavaScript 一些基本语法的用法。alert()、form 对象，以及一些事件的用法将在后面的章节中详细介绍。读者要特别注意 HTML 不区分大小写，而嵌在其中的 JavaScript 却是严格区分大小写的，一旦混淆，将给开发带来不必要的麻烦。

【运行程序】运行上述代码，当输入“日期=1”和“员工号=123”时，单击“递交”按钮，运行结果如图 13.8 所示。



图 13.8　员工号不是 6 位的运行结果

【深入学习】单击“确定”按钮，重新输入“日期=1”和“员工号=12345d”时，单击“递交”按钮，运行结果如图 13.9 所示。在图 13.9 的对话框中单击“确定”按钮，再重新输入“星期=2”和“员工号=123456”，单击“递交”按钮，运行结果如图 13.10 所示。



图 13.9 员工号并非全是数字时运行结果



图 13.10 员工号输入正确时运行结果

单击“确定”按钮，弹出对话框如图 13.11 所示。



图 13.11 返回员工是第几位访客

经过上述例子的演示，JavaScript 的特效更进一步地显现出来了。在程序 13.4 中涉及了变量、基本数据类型、数据类型的转换方法、运算符、表达式、流程控制（if 结构、for 结构），以及函数的用法。笔者建议初学者照着例子自己运行一下，结合运行结果和代码更进一步地体会 JavaScript 基本语法的定义和使用。

13.8 小　　结

通过本章的学习，读者对基本的程序语法有了简单的概念，也许目前还不能很好地体会这些语法究竟能做什么，不过了解这些基础会对今后的学习有更好的帮助。本章是 JavaScript 的入门部分，主要

介绍了脚本语言、VBScript 和 JavaScript 的概念，并通过例子初步展现了脚本语言的魅力。JavaScript 很容易使人望文生义，不自觉地就与 Java 联系起来，但是 JavaScript 与 Java 有着很大的不同，区分这两个概念可以使初学者避免一些错误。

JavaScript 基本语法是本章的主要内容，包括标识符、基本数据类型、运算符、表达式、流程控制语句和函数。通过简单的几行代码来分别介绍这些语法，最后通过“员工工作时间查询系统”这个案例集中展示了 JavaScript 基本语法的用法。在后面的章节中将开始介绍表单在页面中的效果，表单是商业网站的重要部分。

第 14 章　JavaScript 入门

将 JavaScript 引入 HTML 的目的在于实现客户与浏览器的动态交互，第 13 章已经初步体验到了其效果。JavaScript 是基于对象的脚本语言，即所有的编程均以对象为出发点。把 JavaScript 中的元素划分给大大小小的对象，对象中仍然可以包含对象。通过学习将更进一步体验 JavaScript 的特效。例如，在描述银行系统时，银行员工具有名字、职位、考核成绩等属性，同时还必须包括员工可以执行的存款、核算、打印单据等操作，在设计网页时，要访问每一个员工的属性和操作该怎么实现呢？本章将详细阐述。本章的知识点如下。

- ❑ 对象的概念、DOM 是什么。
- ❑ JavaScript 中的一个重要数据结构——数组。
- ❑ JavaScript 中常用的内部对象。
- ❑ window 对象的属性、方法和事件。
- ❑ document 对象的属性、方法和事件。
- ❑ 一个 JavaScript 实现的动态页面。

14.1　了解一下何为“对象”

“对象”这个词相信大家都不陌生，在很多计算机语言中都有这个概念，所以才称这些编程语言为面向对象的，表示这是一类事物。在生活中，如果你是一个销售商，你把你的产品卖给你的客户，那么你的客户就是你的销售对象，其实这与编程语言中的“对象”颇有相似性。JavaScript 中的对象是指 JavaScript 这门语言所服务的一类事物。

14.1.1　JavaScript 对象概述

JavaScript 中的对象与面向对象编程语言中的类的概念相似，是对一类事物的描述。但与面向对象的编程语言不同的是，JavaScript 对象没有抽象、继承、重载等功能。JavaScript 中的对象一般包括属性和方法两个基本元素。对象的属性是反映对象某些特定性质的，是信息的装载单位，可以理解为变量。例如，窗口的大小、文字的颜色等。而对象的方法是指对象可以执行的动作，这些动作能够按照设计者的意图被执行，可以理解为函数。例如，提交表单、鼠标单击的处理函数等。

对象是一个抽象的概念，将其具体化之后就是对象实例，也就是说对象与对象实例是一般与具体的关系。例如，“汽车”表示一类有发动机、四轮的事物，是一个对象，而“奥迪 A6”表示一种特定型号的汽车，是一个对象实例。

那么对象和对象实例是怎么联系的呢？答案是构造函数。构造函数是用来创建对象实例的函数。在定义对象时可以自己定义构造函数，如果没有定义解释器，则会默认定义一个构造函数，如以下这

个例子：

```
var objectInstance=new objectName([参数列表]) ;
```

其中，objectInstance 表示将要创建的对象实例的名字，objectName 则是对象名字，参数列表是创建对象实例时传递的参数，[]表示可以选择，参数的个数是零个或多个。

当创建了对象实例之后，就可以访问对象实例的属性和方法。最常用的访问方式是在对象实例后面加上一个点（.）和一个成员名（属性或者方法）。如果对象实例后面跟的成员名没有定义过，浏览器执行时应当作为这个对象实例新增一个成员。这就是 JavaScript 的特殊之处，可以无限地为对象实例添加新成员。另外，在访问对象实例的属性时可以采用格式“对象实例名[属性名字符串]”。这种格式可以实现对对象属性动态访问的效果。如程序 14.1 中的定义及访问对象实例。

【本节示例参考：资料光盘\第 14 章\14-1　定义及访问对象实例.html】

【实例 14-1】定义及访问对象实例，其源码展示如下：

程序 14.1　定义及访问对象实例.html

```
01	<script language="javascript">
02	function Employee()			//定义构造函数
03	{
04	}
05	var employee=new Employee();		//初始化对象实例 employee
06	employee.name="LiMIng";			//给 employee 新增一个成员变量 name
07	employee.id="031256";			//给 employee 新增一个成员变量 id
08	function query()
09	{
10	alert(employee.name+"->"+employee.id);
11	}
12	employee.queryInfo=query;		//给 employee 新增一个方法
13	employee.queryInfo();
14	</script>
```

【运行程序】浏览该页面，结果如图 14.1 所示。

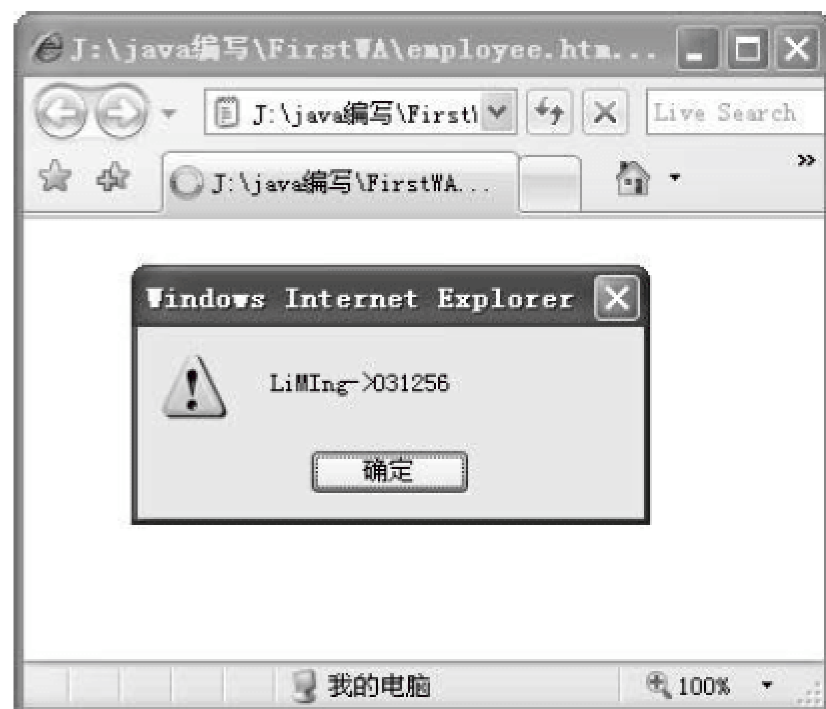


图 14.1　定义及访问对象实例

【深入学习】实例 14-1 实现的内容是：

（1）定义一个构造函数 Employee()，这一步相当于定义好了对象。

（2）使用 new 关键字就可以来创建对象实例 employee。

（3）为对象实例 employee 添加 name 和 id 两个属性，并添加一个方法 query()。

（4）调用对象实例 employee 的方法。

如图 14.1 所示，可以看到浏览器在解释该段程序时调用了 queryInfo()方法，这就表明代码成功地定义和访问了对象。

14.1.2 DOM 介绍

在 JavaScript 中有许多对象，设计者通过什么方式来组织这些对象呢？这就需要了解一个新概念——DOM。

DOM（Document Object Model）即文档对象模型，它是 W3C 的标准。它的功能是把浏览器支持的文档（包括 HTML、XML、XHTML）当作一个对象来解析。DOM 实际上是一个操作文档中所包含的一个编程的 API，允许开发人员从文档中读取、搜索、修改、增加和删除数据。



说明：DOM 是与平台和语言无关的，就是说只要是支持 DOM 的平台和编程语言，都可以用来编写文档。

DOM 里面有专门的 HTML 和 XML 的对象模型。用它们来操作文档元素非常方便。DOM 可以视为一种 API 的应用。也就是说，将文件视为一个文件对象，通过程序语言调用 DOM 对象，来对该文件的某些特定数据进行访问操作。并且利用程序将获取的对象数据做更进一步的应用。可以利用 DOM 方法和属性去通过语言（如 VBScript、JavaScript、ASP）操作 XML 文件。

简单地说，DOM 是指对象及对象之间的层次关系。可以通过简单的示例来了解 DOM 的概念。汽车、车窗、玻璃这几个对象，玻璃附属于车窗，而车窗附属于汽车，类似如此，各个对象有着层次关系。JavaScript 中文档对象的结构如图 14.2 所示。



图 14.2 文档对象结构图



注意：在图 14.2 中可以清楚地看到文档对象中各对象之间的并列关系、包含关系，从而可以知道所学习对象的位置。DOM 是语言无关的 API，这意味着其实现并不与 Java、JavaScript 或者其他语言绑定。然而，鉴于本书的目的，接下来主要集中在 JavaScript 的实现上。

在使用对象时，常常涉及事件、事件驱动和事件处理这几个概念。事件是指通过单击鼠标或者敲击键盘在浏览器窗口或网页元素上执行操作。引起事件的原因称为事件源。例如，单击页面上的“提交”按钮之后，就产生了鼠标单击事件。此处的“提交”按钮就是事件源。事件处理是对象化编程的一个很重要的环节，没有了事件处理，程序就会变得很死，缺乏灵活性。事件的处理程序可以是任意 JavaScript 语句，一般是通过程序或函数对事件进行处理。



注意：图 14.2 中有些对象是全小写的，有些是以大写字母开头的。以大写字母开头的对象表示直接用对象的“名字”（Id 或 Name），或用其所属的对象数组指定。

14.2　JavaScript 中的数组

JavaScript 中的数组是最常用的数据结构之一，是用一个变量来存储一组数据，是一组数据的集合。每个数据是数组的一个元素，每一个数据都有相应的索引，因为数组是严格有序的。索引号以 0 开始，直到数组的 length-1。为了存取数组中任意一个元素，需要采用“数组名[索引号]”的形式。与其他编程语言不同的是，同一个 JavaScript 数组的元素可以是不同的数据类型。

JavaScript 的数组属于核心语言对象，而不是文档对象模型。JavaScript 的数组大小不要求确定。将数据收集到数组中可简化数据管理。例如，通过使用数组，只使用一个参数就可以将一组名称传递给函数。

14.2.1　定义数组和操作数组

JavaScript 中的数组定义方法一般来说有 3 种：一种是匿名的方式，一种是通过 new Array()，另一种是在定义时直接赋值。

匿名方式定义的格式是：

```
var arrA=[];
```

或通过 new Array()定义的格式是：

```
var arrA=new Array();
```

上面这两种定义方式的效果是一样的，就是分配存储空间。在使用时给 arrA 赋值，例如：

```
arrA[0]='2000';
arrA[1]= '2004';
arrA[2]= '2008';
```

此外，定义时直接赋值的写法是：

```
arrA=['2000','2004','2008'];
```

在访问数组中的元素时采用“数组名[索引号]”的形式。

定义数组之后便是操作数组。在 JavaScript 中为数组定义了一些系统方法，可以方便开发者实现操作程序。可以说这些方法是开发者非常喜爱使用的。对于初学者来说，这些方法更是可以达到事半功倍的效果。

- shift()方法：删除原数组第一项，并返回删除元素的值；如果数组为空，则返回 undefined。例如：

```
var arrA = [1,2,3,4,5];            //定义一个数组，包含 5 个元素
var arrB = arrA.shift();           //arrA=[2,3,4,5]   arrB=1
```

- unshift()方法：将参数参加到原数组开头，并返回数组的长度。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.unshift(2008,2012);   //arrA=[2008,2012,1,2,3,4,5]   arrB=7
```

- pop()方法：删除原数组最后一项，并返回删除元素的值；如果数组为空，则返回 undefined。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.pop();                 //arrA=[1,2,3,4]   arrB=5
```

- push()方法：将参数添加到原数组末尾，并返回数组的长度。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.push(2008,2012);      //arrA=[1,2,3,4,5,2008,2012]   arrB=7
```

- concat()方法：返回一个新数组，将参数添加到原数组末尾。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.concat(2008,2012,2016);     //arrB=[1,2,3,4,5,2008,2012,2016]
```

- spice()方法：例如，spice(start,Count,para1,para2,…);表示从 start 开始删除 Count 项，并从该位置起插入 para1、para2 等参数。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.spice(2,2,2008,2009,2010);
var arrB=arrA.spice(0,1);                        //此时运行结果同 shift()一样
arrA.spice(0,0,-2,-1); var arrB=arrA.length;     //此时运行结果同 unshift()方法一样
var arrB=arrA.spice(arrA.length-1,1);            //此时运行结果同 pop()方法一样
arrA.spice(arrA.length,0,6,7);var arrB=arrA.length;     //此时运行结果同 push()方法一样
```

- reverse()方法：将数组反序。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.reverse();              //arrB=[5,4,3,2,1]
```

- sort()方法：对数组进行排序。例如：

```
var arrA = [6,2,8,3,5];
var arrB = arrA.sort();          //arrB=[2,3,5,6,8]
```

- slice()方法：返回从原数组中指定开始下标到结束下标之间的项共同组成的新数组。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.slice(2,3);       //arrB=3
```

- join()方法：将数组的元素组成一个字符串，以 sign 为分隔符，省略则默认用逗号为分隔符。例如：

```
var arrA = [1,2,3,4,5];
var arrB = arrA.join("->");       //arrB= "1->2->3->4->5";
```

下面来展示一下以上这些方法在页面中实际应用的效果，如程序 14.2 的页面。

【本节示例参考：资料光盘\第 14 章\14-2　数组方法.html】

【实例 14-2】数组方法，其源码展示如下：

程序 14.2　数组方法.html

```
01        <script language="javascript">
02        var arrA = [1,2,3,4,5];                              //定义一个数组，包含 5 个元素
03        document.write("数组 arrA=["+arrA+"]"+"<br>");     //读取该数组
04        var arrB = arrA.shift();
05        document.write("shift()方法的效果:arrA=["+arrA+"],删除的元素："+arrB+"<br>");
06        var arrC = arrA.unshift(2008,2012);
07        document.write("unshift()方法的效果:arrA=["+arrA+"],数组长度："+arrC+"<br>");
08        var arrD = arrA.pop();
09        document.write("pop()方法的效果:arrA=["+arrA+"],删除的元素："+arrD+"<br>");
10        var arrE = arrA.push(2008,2012);
11        document.write("push()方法的效果:arrA=["+arrA+"],数组长度："+arrE+"<br>");
12        var arrF = arrA.concat(2008,2012,2016);
13        document.write("concat()方法的效果:arrF=["+arrF+"]"+"<br>");
14
15       var arrT = [1,2,3,4,5];                               //定义一个数组，包含 5 个元素
16        document.write("arrT=["+arrT+"]"+"<br>");
17        var arrG = arrT.reverse();                           //将数组元素倒序排列
18        document.write("reverse()方法的效果:arrG=["+arrG+"]"+"<br>");
19        var arrH = arrT.sort();                              //将数组排序
20        document.write("sort()方法的效果:arrH=["+arrH+"]"+"<br>");
21        var arrI = arrT.slice(2008,2009);
22        document.write("slice()方法的效果:arrI=["+arrI+"]"+"<br>");
23        var arrJ = arrT.join("->");                          //将数组元素组成一个字符串，以"->"分割
24        document.write("join()方法的效果:arrJ=["+arrJ+"]"+"<br>");
25        </script>
```

【运行程序】浏览该页面，运行结果如图 14.3 所示。

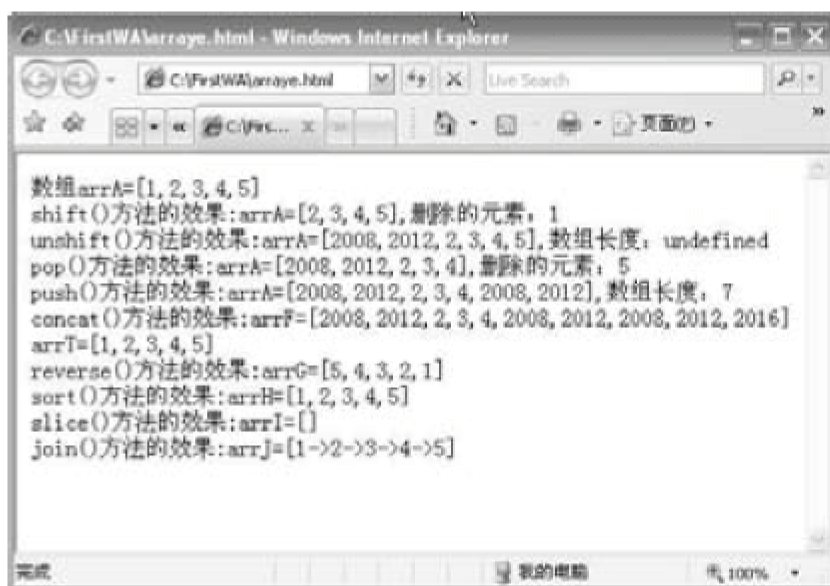


图 14.3　数组方法演示结果



说明：通常情况下，排序只需要使用 sort()方法即可。删除数组的头元素和最后一个元素等，使用数组的系统方法即可。

14.2.2　多维数组

JavaScript 中严格地讲没有多维数组，然而这种数据结构是非常需要的。在开发过程中可以通过数组嵌套来实现，即数组元素本身又是数组。例如，管理一组奥运志愿者的信息，小组包括 5 名志愿者，而每名志愿者的信息包括姓名、年龄、籍贯、学校、所属小组和编号等。此时就可以用嵌套数组来定义。格式如下：

```
数组变量名[子数组索引号][子数组中的元素索引号]
```

这是一个嵌套数组的例子。

```
var arr=[["zhangsan",23,"031256"],["Lisi",22,"031266"],["Wangwu",21,"031245"]];
```

这个数组中包含 3 个子数组，每个子数组描述个人的信息，包括姓名、年龄、编号。通过 arr[i]（i=0,1,2）来访问任何一个子数组，而通过 arr[i][j]（i,j=0,1,2）来访问子数组中的任何一个元素。



说明：本书对于 JavaScript 的介绍仅是泛泛而谈，为的是令读者从 HTML 语言更好地过渡到学习 JavaScript。如果读者希望了解更多的使用 JavaScript 的技巧，需要查阅相关的书籍。

14.3　内 部 对 象

JavaScript 中有一些常用的内部对象（也称核心对象），在引用某些对象的属性和方法时不需要使用 new 关键字来创建对象实例，而是可以直接采用“对象名.成员”的格式来访问。这样的对象称为静态对象，与之对应，需要用 new 关键字来创建对象实例的对象，称之为动态对象。常用的内部对象有 Math 对象、Date 对象和 String 对象。

14.3.1　Math 对象

JavaScript 中提供了一些数学工具，可以大大地节省开发者的时间，而无须将精力耗费在求绝对值、平方根，以及三角函数值等数学处理上。这个对象与 JavaScript 中的其他对象不同，开发者无须通过 new 来生成对象实例。Math 是静态对象，直接通过对象引用相应的属性和方法即可。

Math 对象的属性通常是常数，其使用的方法很多，包括三角函数、反三角函数，还有一些预定义的数学函数。

14.3.2　Date 对象

生活中，人们通过月历年历等方法计数“准确的某年某月某日”。在 JavaScript 中，有专门这样的一个对象用来描述日期和时间。这个对象是 Date 对象，它的日期和时间是按照 GMT（即格林威治时间）来计量的。在计算机内部日期时间是一个整数，从 1970 年 1 月 1 日 0 点 0 分 0 秒起相对某个日期时间的以毫秒为单位的数值，通过这个数值可以计算出其相应的具体日期时间。

Date 对象的最简单的构造函数是 Date()，格式写法是：

```
var today=new Date();
```

上面的方法定义了 Date 对象的实例，也可以通过构造函数来表示过去或者将来的某个时间。例如：

```
var past=new Date(2000,1,1);                //创建一个对象实例，值是 2000 年 1 月 1 日
var tomorrow=new Date(2012,6,6);            //创建一个对象实例，值是 2012 年 6 月 6 日
```



注意：此处有个问题容易出错，past 的值是 Tue Feb 1 00:00:00 UTC+0800 2000，tomorrow 的值是 Fri Jul 6 00:00:00 UTC+0800 2012，显示的结果是参数中月份+1。

14.3.3　String 对象

String 对象是动态对象，需要通过 new String()创建对象实例来调用属性和方法。事实上任何一个字符串常量都是一个 String 对象，可以直接采用“字符串常量.属性（方法）”的格式来调用 String 对象的属性和方法。



注意：这两种调用方法的区别，可以采用 typeof()来体现：

```
typeof(new String("Beijing2008"));
typeof("Beijing2008");
```

前者返回的是 object 类型，而后者返回的是 string 类型。

常用的 String 对象属性有 constructor 属性和 length 属性。

- constructor 属性：表示创建对象的函数。例如：

```
stringA= new String("Beijing2008");          //创建一个字符串对象实例
if(stringA.constructor==String)              //检查是否创建了一个字符串对象实例
alert(stringA);
```

由运行可知，判断条件为真，执行了弹出对话框显示字符串的操作。另外一个属性是 length 属性，它是 String 对象最常用的属性之一。

- length 属性：返回字符串的长度，例如：

```
stringA= new String("Beijing2008");        //创建一个字符串对象实例
leng=stringA.length;
```

或者也可以写为：

```
stringA="Beijing2008";                     //创建一个字符串对象实例
leng=stringA.length;
```

通过使用这两种定义方式分别来调用 String 对象的 length 属性。由运行结果可知，两者返回的结果一样，都是 11。



说明：使用 String 对象的方法很多，其作用包括用于查找和匹配字符串中字符的方法，或者是用于提取子字符串的方法和改变字符串大小写的方法等。

14.4 window 对象

window 对象对于文档对象模型的最高层，代表浏览器的整个窗口，是最大的一个对象，因为所有的事件都发生在窗口中。开发者可以通过 window 对象的属性、方法和事件处理来控制浏览器的显示效果。不必专门在 JavaScript 代码中创建 window 对象。当打开浏览器以后会自动打开一个窗口，这个窗口就是一个可用的 window 对象。在调用其属性和方法时，不需要用 window.xxx 的格式，可以省略 window 这个前缀。

14.4.1 window 对象属性

在 JavaScript 语言中，window 属性有很多，接下来通过一个实例来观察这些属性的效果。这其中包含 name 属性、closed 属性和 opener 属性。

- name 属性的作用是设置或获取表明窗口名称的值。
- closed 属性的作用是获取引用窗口是否已关闭。
- opener 属性的作用是设置或获取创建当前窗口的引用。

具体在案例中的使用如程序 14.3 和程序 14.4 的页面，这是应用 window 对象的 JavaScript 程序。

【本节示例参考：资料光盘\第 14 章\14-3　window 对象的主页面.html】

【实例 14-3】window 对象的主页面，其源码展示如下：

程序 14.3　window对象的主页面.html

```
01  <head>
02  <script>
03          var pop;
04          pop= window.open
05            ("child.htm","pop","width=255,height=235,resizable=1,scrollbars=auto,toolbar=no,
06          menubar=no,location=0,top=10,left=200");      //打开子页面 child.htm
07          window.name="测试 Opener 属性";
08          alert(window.screenLeft+":"+window.screenTop);
09          function b1(){                //如果子窗口还没有关闭，则关闭它
10            if(!pop.closed)
11              {
12                pop.close();
13              }
14          }
15        </script>
16  </head>
17        <body>
18          window 属性示例<p>
19          <font color=red onmouseover="javascipt:window.status='鼠标指向我';"
20        onmouseout="javascipt: window.status='鼠标没有指向
21          我';">把鼠标移动这里来</font> <br>          <!--触发鼠标事件-->
22          <input type="button" name="b1" value="关闭窗口" onclick="javascript:b1();">
23        </body>
```

【本节示例参考：资料光盘\第 14 章\14-4　child.html】

【实例 14-4】window 对象的子页面，其源码展示如下：

程序 14.4　child.html

```
01  <html>
02        <body>
03          这是一个测试文件，用于测试文件的打开与关闭。
04          <p><font color=red onclick="javascipt: alert(window.opener.name);">单击显示父窗口
05          的名称</font>        //触发 onclick 事件
06         </body>
07  </html>
```

【运行程序】浏览该页面，上边代码中有两个文件，主页面实现的功能是：

（1）打开一个新页面，并通过一些属性设置新页面。

（2）如程序 14.3 中第 7 行，通过 window.name 设置窗口的名称为“测试 Opener 属性”。

（3）同时弹出对话框显示浏览器客户区左上角相当于屏幕左上角的 x、y 坐标。

（4）通过鼠标移动事件检查浏览器状态栏的变化。

（5）关闭子窗口功能，通过 closed 属性检查子窗口是否已经关闭。而子页面实现的功能是：显示父窗口的名称。

实例 14-3 运行的结果如图 14.4 所示。

【深入学习】 首先浏览器中能看到弹出的对话框，对话框中显示的坐标值是对话框相当于屏幕左上角的 x、y 坐标，并且弹出子页面，它的作用是测试文件的打开和关闭。如图 14.5 所示，当单击子页面中红色文本“单击显示父窗口的名称”时，页面弹出对话框表明主页面的名称为“测试 Opener 属性”。

当单击主页面的“确定”按钮后，如图 14.5 所示。如果鼠标放置的位置没有在红色文本字体上，则状态栏显示“鼠标没有指向我”；反之，如果鼠标移动到红色文本字体上时，则页面显示“鼠标指向我”。



图 14.4　主页面显示效果



图 14.5　子页面显示效果

此时再单击主页面的“关闭窗口”按钮，即图 14.6 所示的效果，单击“关闭窗口”按钮，子窗口就被关闭。这体现了 closed 属性的效果。



图 14.6　单击“确定”按钮后的显示结果

说明：window 对象的属性还有很多，如表 14.1 所示。

表 14.1　window对象的属性

属　性　名	描　　述
defaultStatus	设置或获取要在窗口底部的状态栏上显示的默认信息
dialogArguments	设置或获取传递给模式对话框窗口的变量或变量数组
dialogHeight	设置或获取模式对话框的高度
dialogLeft	设置或获取模式对话框的左坐标
dialogTop	设置或获取模式对话框的顶坐标

续表

属　性　名	描　　述
dialogWidth	设置或获取模式对话框的宽度
frameElement	获取在父文档中生成window的frame或iframe对象
length	设置或获取集合中对象的数目
offscreenBuffering	设置或获取对象在对用户可见之前是否要先在屏幕外绘制
opener	设置或获取创建当前窗口的引用
returnValue	设置或获取从模式对话框返回的值
screenLeft	获取浏览器客户区左上角，相当于屏幕左上角的 x 坐标
screenTop	获取浏览器客户区左上角，相当于屏幕左上角的 y 坐标
self	获取对当前窗口或框架的引用
status	设置或获取位于窗口底部状态栏的信息
top	获取最顶层的祖先窗口

14.4.2　window 对象方法

所谓方法即是如何通过 window 对象属性来使这些属性发挥各自的作用。和属性一样，window 对象属性所对应的使用方式也有很多。本小节通过一系列不同属性的使用方法，来了解一些常用 window 对象的属性调用。

- open()、close()方法：产生新窗口的方法就是 window.open()。例如：

```
01    var newWindow1=window.open("world.html","world","height=300,width=400");
02    var newWindow2=window.open("../world.html","we","height=100, resizable=no");
03    var newWindow3=window.open
04    ("http://www.sohu.com/","sohu","height=100,width=200,resizable=no,
05    menubar=yes,toolbar=no");
```

上述例子定义了变量（如 newWindow1），当运行 window.open()方法时将新窗口的引用赋给变量 newWindow1，之后就可以通过引用 newWindow1 调用方法来控制新窗口。open()方法有 3 个参数，第一个参数是要打开的页面的 URL，当新窗口与主窗口在同一目录下时，直接写文件名即可，如上述代码第 1 行。当新窗口与主窗口不在同一目录下时，则写新窗口的相对路径，如代码第 2 行。当然，新窗口也可以是绝对路径或者是 http 地址，如第 3 个示例。第 2 个参数是新窗口的名称，如果打开新窗口时为空，之后还可通过 window.name 设置。其中第 3 个参数是窗口风格的设置。

close()方法是与 open()方法对应的，来关闭窗口。需要关闭哪个窗口就通过哪个窗口的引用调用 close()方法。在调用 close()方法前一般都会先检查窗口是否被打开，不然可能会引起不必要的错误。

- alert()方法：弹出一个警告对话框显示参数。单击对话框中的“确定”按钮就可退出对话框。这是一个简单的例子：

```
alert("北京 2008Olympic！One World，One Dream！");
```

这个方法在调试脚本程序时比较有用，有点类似于其他编程语言中的打印语句。



注意：完整的引用格式是“window.alert(字符串);”，但是对于对话框窗口都不引用 window，而直接使用方法，这样简单一些。

- confirm()方法：显示一个具有 OK 和 Cancel 按钮的对话框，根据用户的选择分别返回 true 或者 false，程序根据这个返回值进行不同的处理。例如：

```
var answer=confirm("Are you access the website?");
if(answer)
{
location.href="http://www.sohu.com";        //跳转到 http://www.sohu.com
}
```

当用户单击“确定”按钮时就跳转到 http://www.sohu.com。这个方法可以用在访问可能存在不安全因素的网站时，提示用户选择继续与否。

- prompt()方法：显示一个用户可以输入信息的对话框，并返回用户输入的内容。第 1 个参数是提示信息，以字符串形式表达。第 2 个参数是输入信息的默认值，可以提供输入信息的样本。例如：

```
var answer=prompt("what is your name?","");
if(answer)
{
alert("Hello!"+answer+",WELCOME!");
}
```

下面将上述这些使用方法整合在同一个页面中，这样，页面将展示出独特的魅力，如程序 14.5 的 window 对象页面。

【本节示例参考：资料光盘\第 14 章\14-5　使用 window 对象方法的主页面.html】

【实例 14-5】使用 window 对象方法的主页面，其源码展示如下：

程序 14.5　使用window对象方法的主页面.html

```
01            <script language="javascript">
02              var pop= window.open
03              ("child.html","pop","width=255,height=235,resizable=1,scrollbars=auto")
04                  pop.setTimeout("close()",2*5000);   //打开子页面 child.html
05             function method()           //此方法展示 confirm 的用法
06             {
07                  alert("北京 2008Olympic！ One World，One Dream！ ");
08                  var answer=confirm("Do you want to see news about 2008Olympic?");
09                  if(answer)
10                   {
11                        var yourName=prompt("what is your name?","");
12                        if(yourName)
13                        {
14                             alert("Hello!"+yourName+",WELCOME!");
15                        }
16                   }
17                setInterval("changeWindow()",2000); //每隔 2 秒调用方法 changeWindow()
18             }
```

```
19              function changeWindow()
20                  {
21                      window.resizeBy(-10,-10);       //改变窗口大小
22                  }
23              </script>
24              <body bgcolor="white" onload="method()">
25                  <H1>window 方法示例</H1>
26                  <H2>子窗口 child.html 在打开 10 秒钟后就关闭！ </H2>
27              </body>
```

【本节示例参考：资料光盘\第 14 章\14-6　child.html】

【实例 14-6】使用 window 对象方法的子页面，其源码展示如下：

程序 14.6　child.html

```
01              <html>
02                  <body>
03                      <center>
04                          <font color="red"><H1>我 10 秒钟后就关闭，珍惜我噢!</h1></font>
05                      </center>
06                  </body>
07              </html>
```

【运行程序】上面程序 14.5 包含使用 window 对象方法的主页面，程序 14.6 是使用 window 对象方法的子页面，这两段代码共同完成了这个功能。当浏览者打开这个页面时，首先出现的是如图 14.7 所示的页面。

在载入页面时弹出警告对话框“北京 2008Olympic！ One World，One Dream！ ”，单击“确定”按钮后，显示效果如图 14.8 所示。

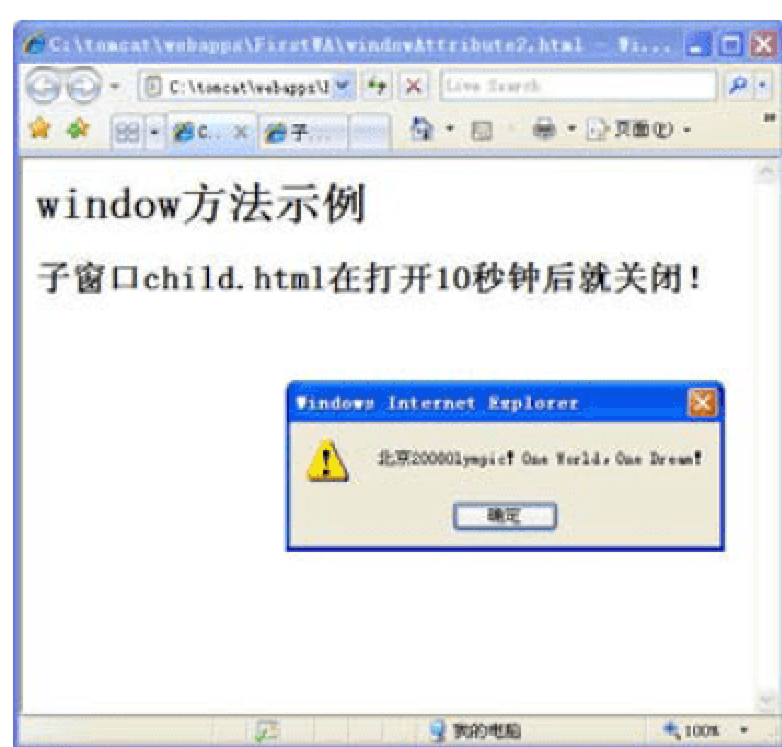


图 14.7　打开 windowMethod.html 的效果

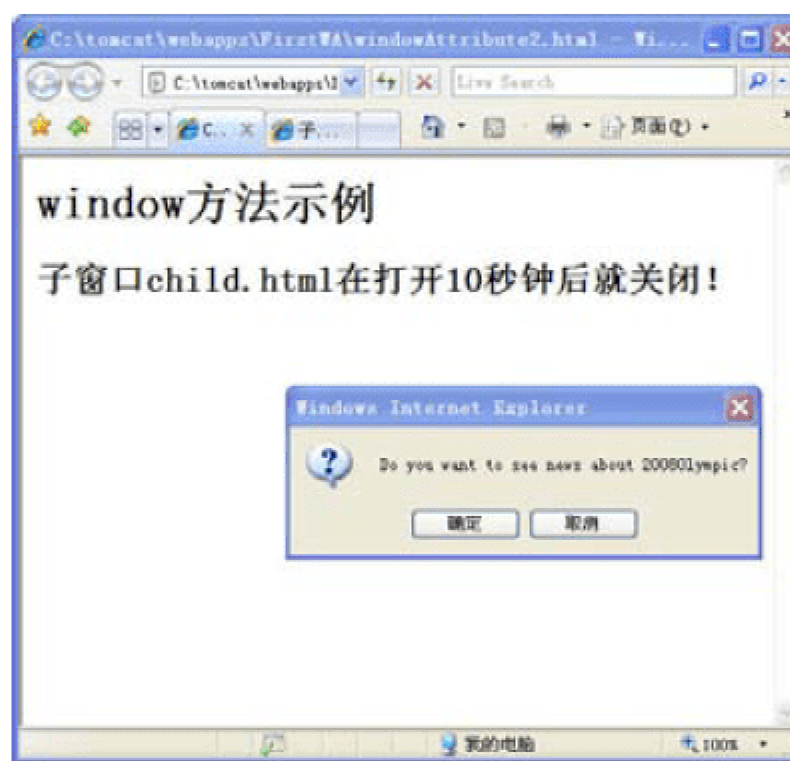


图 14.8　确认对话框的显示效果

弹出确认对话框，显示提示信息“Do you want to see news about 2008Olympic?”。当浏览者单击“确定”按钮后，显示效果如图 14.9 所示。

【深入学习】如果在确认对话框中单击“确定”按钮，则弹出可以输入信息的对话框，提示“what

is your name?"。如果输入不为空，则通过对话框显示用户的输入信息。例如，在文本框中输入字符串"Zhang Huiji"，单击"确定"按钮，显示效果如图 14.10 所示。



图 14.9　可以输入信息的对话框



图 14.10　显示文本框中输入的字符串

单击"确定"按钮后，窗口每隔 2 秒在水平和垂直方向均缩小 10px。直到浏览器客户区完全消失后，如图 14.11 所示。



图 14.11　窗口逐渐减小效果

从图 14.11 中可以看到，窗口中的字体相对于窗口来说逐渐变大，这就是由于窗口逐渐减小的缘故。此外，在打开主页面窗口的同时，也自动打开子页面窗口，这个窗口存在 10 秒后也会自动关闭。

14.4.3　事件

在操作浏览器时，大多数动作都是通过鼠标操作引起的，另一些通过敲击键盘引起，因为鼠标和键盘是日常生活中最常用的输入设备。那么用户与浏览器交互怎么实现？输入设备引起的事件怎么处理？什么时候处理？这就要求助于事件和事件处理程序了。window 对象的事件是最令人入迷的部分之一。没有事件处理程序就会显得死板，不够灵活。

window 对象的事件处理程序要放在<body>标签的时间属性设置中，例如：

```
onload="window_onload()
```

onload 就是一个事件处理程序。类似这样的事件很多，通过一系列的整合，可以实现很多实用的功能，如程序 14.7 提交注册号的页面。

【本节示例参考：资料光盘\第 14 章\14-7　window 对象的事件.html】

【实例 14-7】window 对象的事件，其源码展示如下：

程序 14.7　window对象的事件.html

```
01          <script language="javascript">
02            var pop;
03            function window_onload()                  //此函数打开子窗口 child.html
04            {
05              pop=
06          window.open("child.html","pop","width=255,height=235,resizable=1,scrollbars=auto" );
07           }
08           function window_onkeypress(){
09              if(window.event.keyCode==27)          //按 Esc 键退出页面
10                window.close();
11              else
12                alert(window.event.keyCode);        //显示 ASCII 码
13           }
14           function checkNum(){
15              num.value="031268";                   //初始号码设为 031268
16           }
17           function window_onclick(){
18              alert("提交成功");                     //触发后弹出提交成功
19           }
20           function child_close(){
21              pop.close();
22           }
23           </script>
24           <body onload="window_onload()" onkeypress="window_onkeypress();"
25           onbeforeunload="window.event.returnValue='子页面同时也会被关闭'"
26         onunload="child_close()">
27         你的编号：<input type=text name=num onfocus="checkNum()"><br>
28         <input type=submit name=submit1 value="递交" onclick="window_onclick()">
29          </body>
```



注意：open 函数中的 child.html 文件是一个已存在文件，内容无所谓，此处只是为检查 onload 的用法。

【运行程序】上述代码实现的功能过程是，在页面载入打开后，如图 14.12 所示。打开主页面后，触发 onload 事件打开子窗口，当鼠标单击文本框时触发 onfocus 事件，文本框中自动显示“031268”。然后单击“递交”按钮，触发 onclick 事件，弹出警告框显示“提交成功”，如图 14.13 所示。

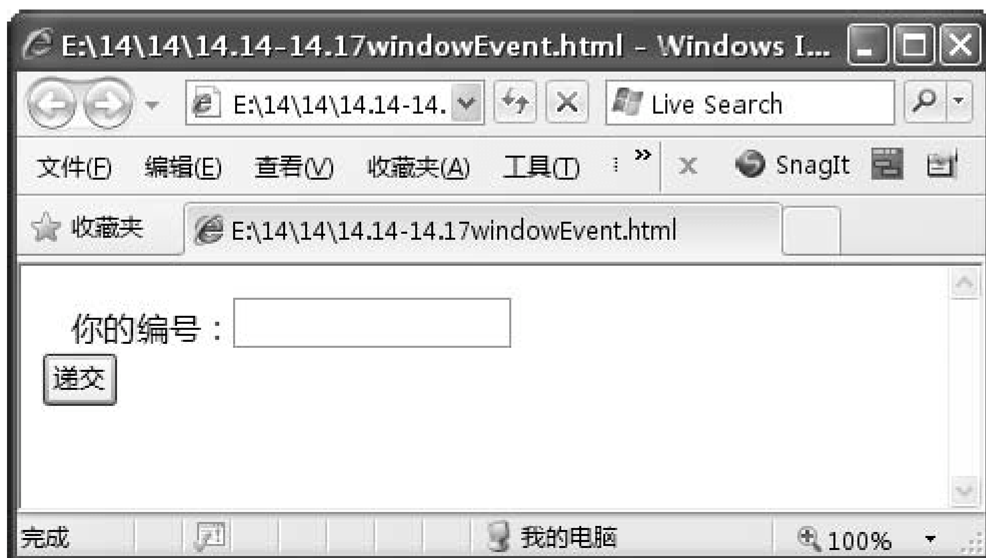


图 14.12 windowEvent.html 显示效果

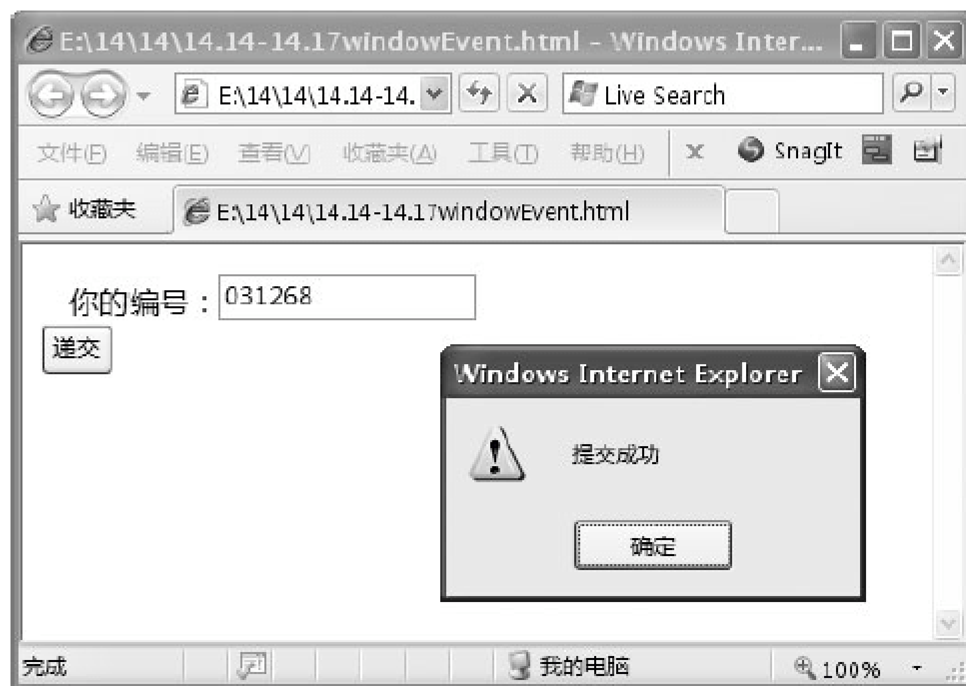


图 14.13 触发 onfocus 和 onclick 事件的效果

【深入学习】这个页面还有一个有意思的功能是，如果浏览者单击其他任意键弹出对话框，页面将显示其 Unicode 码，如图 14.14 所示，如按下键盘上的 G 键。

单击“确定”按钮后，如果用户关闭此窗口，将触发 onbeforeunload 事件，弹出确认消息，如图 14.15 所示。如果单击“确定”按钮后，触发 onunload 事件，在主窗口关闭的同时也关闭子页面。

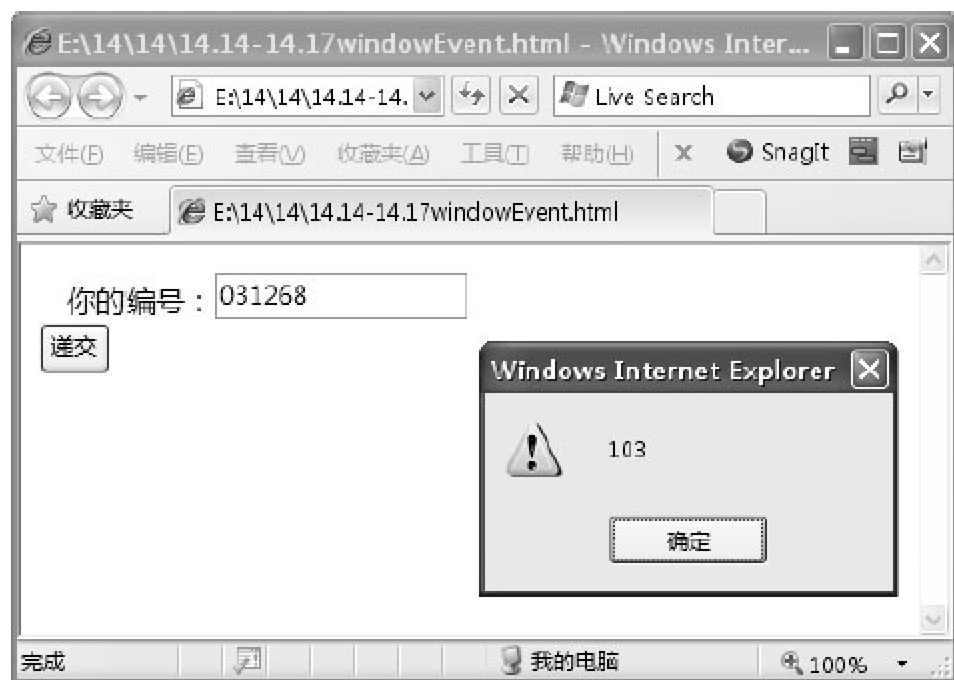


图 14.14 显示键盘的 Unicode 码

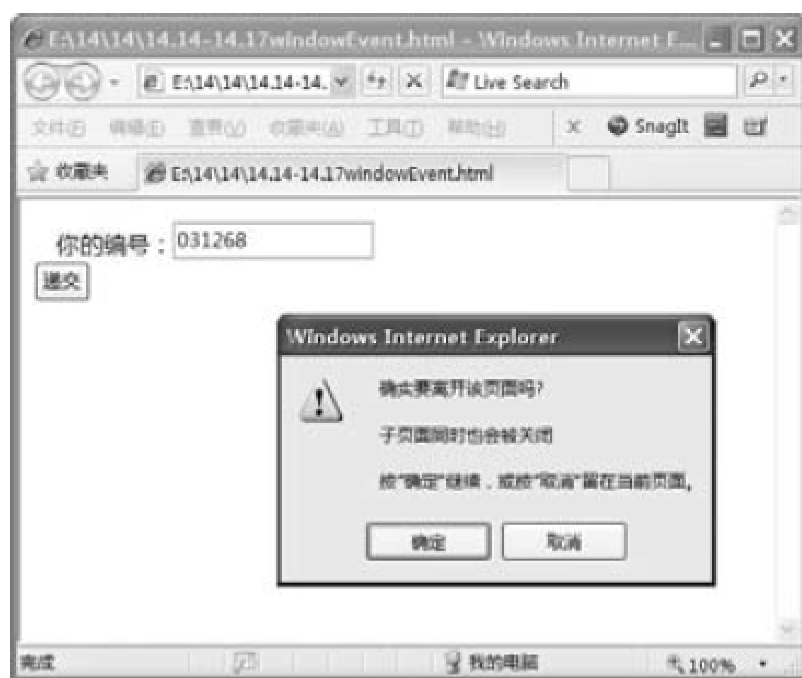


图 14.15 触发 onbeforeunload 事件的效果



说明：通过这个例子读者可以了解 window 对象事件。当然，window 对象事件有很多，读者可以参考更多书籍。

14.5 document 对象

document 对象是 window 对象的一个对象属性，代表浏览器窗口中装载的整个 HTML 文档。文档中的每个 HTML 元素都对应着 JavaScript 对象。设计者要结合 HTML 标签和 JavaScript 产生最佳效果。因为 document 代表整个 HTML 文档，是其他 HTML 元素的根节点，其他节点可以通过 document 引用。document 的属性和方法一般用来设置页面文档的外观和内容。

14.5.1 document 对象属性

document 对象属性实现的功能使用 HTML 中的标签也可以实现，从其表面效果来看是差不多的，但是与事件处理程序结合使用时就体现出了两者的不同。常用的属性如表 14.2 所示。

表 14.2 document对象的属性

属 性	说 明
title	设置文档标题
bgColor	设置页面背景色
fgColor	设置前景色，即文本颜色
linkColor	未访问过的链接颜色
alinkColor	在超链接上单击鼠标时的颜色
vlinkColor	已经访问过的链接颜色
URL	设置URL属性从而在同一窗口打开另一网页
fileCreatedDate	以字符串形式返回文件建立日期
fileModifiedDate	以字符串格式返回文件修改日期
filesize	返回网页文档的大小
cookie	设置和返回cookie值
charset	返回网页文档所使用的字符编码方式

14.5.2 document 对象方法

document 对象的方法最常用的就是 writeln()方法，如表 14.3 所示是一些常用的 document 对象方法。

表 14.3 document对象的方法

方 法	描 述
open	打开一个流，和window对象的open方法类似
close	关闭open()方法打开的输出流
write	向HTML文档中写入内容
writeln	与write()方法相比，每次写完内容之后多一个换行符
getElementById()	返回文档中任何元素（ID属性具有唯一性）的引用
getElementsByName()	返回指定name属性值的对象的引用数组
getElementsByTagName()	返回指定标签名的对象的引用数组
createElement	产生一个代表某个HTML元素的对象，而后使用父元素的方法来修改文档的内容（如appendChild()方法）

14.5.3 document 对象事件

document 对象常用的事件处理程序就是 window 对象中介绍的通用事件，此处不再赘述。下面通过一个 document 对象的例子来演示其属性和方法的使用。

【本节示例参考：资料光盘\第 14 章\14-8 document 对象的属性和方法 document.html】

【实例 14-8】document 对象的属性和方法 document.html，其源码如程序 14.8 所示。

程序 14.8　document.html

```
01          <script language="javascript">
02              function setDocument()
03              {
04                  document.bgColor="white";                        //设置背景颜色为白色
05                  document.fgColor="black";                        //设置字体颜色为黑色
06                  document.alinkColor="red";
07                  document.vlinkColor="green";
08                  document.linkColor="yellow";
09              }
10            function create_Element()
11            {
12                var area = document.getElementById("area");       //通过"area"取得对象
13                var element = document.createElement("input");    //动态创建一个对象
14                element.type = "radio";                           //对象类型是单选按钮
15                      var obj = area.appendChild(element);        //将对象插入到 area 中
16                obj.checked = true;                               //单选按钮默认状态是选中状态
17            }
18          </script>
19          <body onload="setDocument()">
20              <H1>document 对象的属性和方法示例</H1><br>
21              <font color=red onclick="javascript:create_Element()">点一下这里</font><br><br>
22              <a href="newWindow.html">进入新窗口</a><br>
23              <div id="area"></div>
24          </body>
```

这段代码中，document 方法使用到的分别是 getElementById()和 createElement()。getElementById()是指返回文档中任何元素（ID 属性具有唯一性）的引用。所以在代码中，表示将获得 area 层中的元素。createElement()是指产生一个代表某个 HTML 元素的对象，而后使用父元素的方法来修改文档的内容，同样，如 appendChild()方法也是这样。

【本节示例参考：资料光盘\第 14 章\14-9　newWindow.html】

【实例 14-9】链接窗口 newWindow.html，其源码展示如下：

程序 14.9　newWindow.html

```
01          <script language="javascript">
02          document.write("Hello!欢迎来到新窗口");
03          </script>
04          <body>
05          <H1>这是新窗口</H1>
06          </body>
```

【运行程序】上面的程序 14.8 包括 document.html 和 newWindow.html 两个文件。图 14.16 显示了 document.html 打开的效果，有标题、红色文本和超链接。在这里，通过 document 方法设置了网页文档

的背景颜色、字体颜色、超链接等属性信息，而不是通过 HTML 语言。

单击“点一下这里”文本，在“进入新窗口”超链接下出现一个已选定的单选按钮，如单击 6 次“点一下这里”就显示 6 个单选按钮，如图 14.17 所示，这由代码第 14 行和 15 行实现。接下来测试超链接的功能，单击“进入新窗口”超链接，页面转入新窗口，如图 14.18 所示。另外，还可以看到超链接颜色在未访问时、已访问过和选定时的不同。



图 14.16　document.html 显示效果



图 14.17　单击“点一下这里”的显示效果

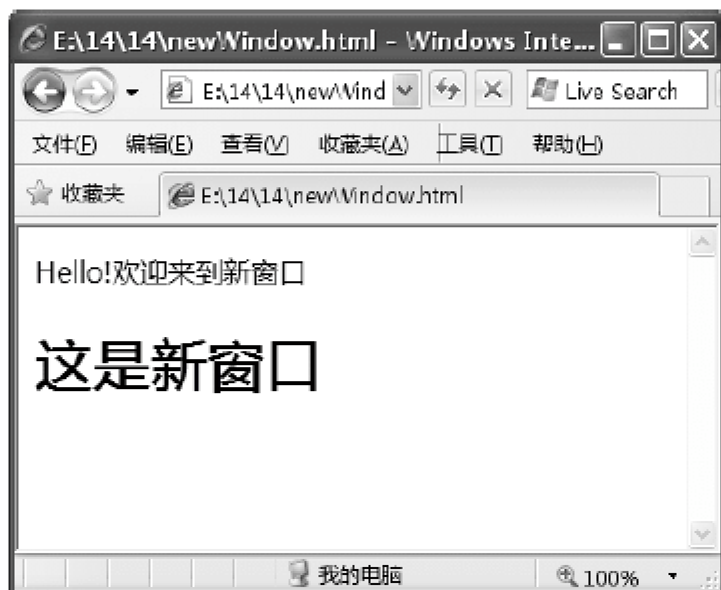


图 14.18　newWindow.html 显示效果

【深入学习】JavaScript 中还有很多其他对象。事实上，HTML 中的每个元素基本上都在 JavaScript 中有相应的对象。由于本书主要是介绍 HTML 和 CSS，JavaScript 部分是想让初学者了解动态页面的实现机理、JavaScript 的基本语法、动态页面一般的编写方法，以配合 HTML 和 CSS 设计出生动活泼的页面来。在实际运用的设计中，这些小例子肯定是无法满足设计需求的，开发者需要将这些基本的知识组合起来，灵活地应用才能达到令人满意的特效。

14.6　案例：一个运用 JavaScript 实现的动态页面

通过前面的学习已经知道，JavaScript 可以用来实现表单验证，例如，登录某个论坛时需要填写用户名和密码；还可以响应鼠标单击事件、键盘敲击事件、窗口移动事件、窗口大小改变事件、文档载入事件和文档卸载事件等。

这就是 JavaScript 能创建动态 HTML 页面，从而实现用户与浏览器交互的原因。使得在有限的页面中展现更多的内容，使网站更加生动，吸引更多的浏览者，给用户良好的体验。在了解了 HTML、

CSS 和 JavaScript 后，就可以动手设计一个漂亮的页面了，这个漂亮除了指外观，还有功能和实用的感觉。来看实例吧，在这个例子中尽可能地包含了前面介绍的对象及其属性、方法和事件。

【本节示例参考：资料光盘\第 14 章\14-10　JavaScript 实现的动态页面 goodLogin.html】

【实例 14-10】JavaScript 实现的动态页面 goodLogin.html，其源码展示如下：

程序 14.10　JavaScript实现的动态页面goodLogin.html

```
01  <html>
02      <head>
03      <title>现在时间是</title>
04      <meta content="text/html; charset=gb2312" http-equiv=Content-Type>
05      <script language=JavaScript>
06      /*在网页标题上显示当前时间*/
07      document.title = document.title+new Date();
08      </script>
09      </head>
10      <body   text="Olive" alink=navy vlink=red link=aqua>
11          <!--图标的区域，此图标可以在页面上移动，下面有函数来实现此效果-->
12      <div id=float_icon style="VISIBILITY:hidden;POSITION:absolute;LEFT=0;TOP=0">
13              <img src="image/2.gif">
14      </div>
15      <div align="center"><center>
16      <font size="4" ><H1>轻松学习 JavaScript</H1></font>
17      <table border=0 cellpadding=0 cellspacing=0><tr bgcolor="gray">
18      <tr><td>
19      <pre><font color="black" size="3">
20      您希望设计出实用、美观的网页吗？你设计的网页美观吗？学
21      习了 JavaScript，您就可以肯定的回答了。
22      </font></pre>
23      </td></tr>
24      </table><br><br>
25      <H2>下面是一个用 JavaScript 实现的登录例子</h2>
26      ————————————————————————————————
27          <!--制作一个表单-->
28      <form name=form1 action="javascript:login()" method=post>
29          用户名：<input type=text name=user onkeypress="checkChar(this.form)"><br>
30          密码：<input type=password name=psw onfocus="checkName(this.form)"><br>
31          <input type=submit name=submit1 value="提交">
32              <input type=reset name=cancel value="取消">
33      </form>
34      </center></div>
35      </body>
36      </html>
```

程序 14.10 是本案例的主体框架，设置该网页的样式和结构。在网页头部实现了显示当前时间的功

能。其中，登录验证函数 checkChar、checkName 和 login 则在下面的 JavaScript 代码中实现。另外，图标在窗体中移动的功能也由程序 14.11 实现。

【本节示例参考：资料光盘\第 14 章\14-11　登录验证代码.html】

【实例 14-11】登录验证代码，其源码展示如下：

程序 14.11　登录验证代码.html

```
01	<script language="javascript">
02			/*下面的代码实现当图标碰到窗口边界时，改变图标移动的方向。
03			如果图标左边位置加上图标宽度大于浏览器窗口宽度，就表示图标
04			已碰到窗口右边界*/
05			 var dirX=1,dirY=1;
06			var xPos=0;yPos=0;
07			float_icon.style.top=0;
08			document.body.all.float_icon.style.left=0
09			document.body.all("float_icon").style.visibility="visible"; //设定图标可见
10			window.setInterval("moveIcon()",100);	//每隔 0.1 秒移动一次位置
11			 /*此函数定义图标的轨迹*/
12			function moveIcon(){
13				xPos+=2*dirX;
14				yPos+=2*dirY;
15				float_icon.style.top=yPos;
16				float_icon.style.left=xPos;
17				if(xPos<=0||xPos+float_icon.offsetWidth>=document.body.clientWidth)
18				   dirX=-dirX;	//如果图标的横坐标超出窗口边界，则图标向相反方向移动
19
20		if(yPos<=0||yPos+float_icon.offsetHeight>=document.body.clientHeight)
21				dirY=-dirY;	//如果图标的纵坐标超出窗口边界，则图标向相反方向移动
22				}
23			/*检查用户名必须是 a-z 的字母*/
24				function checkChar(frm){
25	if(window.event.keyCode>'z'.charCodeAt(0)||
26	window.event.keyCode<'a'.charCodeAt(0))
27				{
28					window.event.returnValue=0;	//取消浏览器的默认事件
29					alert("必须输入 a-z 的字母");
30					}
31				}
32				/*检查必须先输入用户名，再输入密码*/
33				function checkName(frm){
34				if(frm.user.value=="")			//如果用户名为空，则弹出提示信息
35				{
36					alert("你必须先输入用户名");
37					frm.user.focus();
38					}
```

```
39                      }
40                    /*检测是否登录成功*/
41                    function login(){
42                      /*如果用户名为 bupt，密码为 bupt，则登录成功*/
43                      if((form1.user.value=="bupt")&&(form1.psw.value=="bupt"))
44                      {
45                        alert("成功");
46                            }
47                            else
48                                 {
49                                 alert("失败");
50                                      }
51               }
52    </script>
```

【运行程序】在运行时，将程序 14.11 放在程序 14.10 的任意部分都可以，这体现了 JavaScript 的一个特点，因为 JavaScript 代码是放在<script></script>标签对之间，因此不管放在什么位置都具有较好的可读性。代码整合后，使用浏览器打开页面，如图 14.19 所示。



图 14.19　goodLogin.html 显示效果

【深入学习】图 14.19 中的标题上显示着当前时间，有供用户填写用户名和密码的文本框。还有一张笑脸的 GIF 图像浮动在文字上方，并且在各个窗口间不停移动。当碰到窗口边界时，图像会改变移动的方向，如同碰到墙壁被弹开的一样。

在这个页面中，如果用户在“用户名”文本框中输入由字母组成的名字是可以的，但是当输入数字时就弹出警告对话框提示“必须输入 a-z 的字母”，这是通过触发 onkeypress 事件来实现的。当事件被触发时调用 checkChar()函数检查按键是否是字母，如图 14.20 所示。

而如果用户将“用户名”文本框空着，先填密码，会提示用户“你必须先输入用户名”，这是通过触发 onfocus 事件来实现的。当事件被触发时调用 checkName()用户名的值是否为空，如果为空，则提示警告信息，并且“用户名”文本框将得到焦点，此时系统提示必须先输入用户名，如图 14.21 所示。

再次单击“确定”按钮，“用户名”文本框会自动获得焦点。那么如果在程序 14.11 中输入正确的登录名和密码。这里事先将登录用户名设置为 bupt，密码是 bupt，当页面验证两者都正确时则弹出对话框提示成功，如图 14.22 所示，否则提示失败。



14.20　必须输入字母提示窗口



图 14.21　系统提示必须先输入用户名



图 14.22　用户名和密码都为 bupt

在编写 JavaScript 程序时要特别认真，尽量避免函数名错误或者函数名缺失等不必要的错误，尤其是在用记事本编写时，本身没有语法检查的功能，只能在浏览器解释时返回错误。但是浏览器并不能报告 JavaScript 中的所有错误。

例如，在程序 14.10 中，如果将：

```
<input type=text name=user onkeypress="checkChar(this.form)">
```

写成：

```
<input type=text name=user onkeypress="checkChar">
```

结果就比较糟糕。此时，在“用户名”文本框中输入数字时，浏览器不会报告错误，这就违背了设计时的需求。对于这些不易发现的错误，在调试的过程中可以仿照其他编程语言中添加打印语句的方法，在 JavaScript 中添加 alert()语句来找到错误的源头。



说明：对于经验不足的网页设计者来说，这种笨方法是很有效的。当然，在开发者对网页编程较为熟悉时，就会有敏锐的直觉了。大多数情况下，在无法得到预想结果时，熟练的程序员会知道该去查看哪块代码，这就是“熟能生巧”。所以多多练习是必需的。

14.7　小　　结

本章对于 JavaScript 的探讨仅停留在帮助读者从 HTML 到 JavaScript 过渡的层面，是为了帮助读者今后学习 JavaScript 做一个入门的启发。本章介绍的知识点有：

- ❑ 对象和 DOM 的概念。
- ❑ JavaScript 的数组的定义和使用方法。
- ❑ JavaScript 中常用的 3 个内部对象为 Math 对象、Date 对象和 String 对象。
- ❑ 处在文档对象模型的最高层，代表浏览器的整个窗口的 window 对象及其属性、方法和事件，这部分用处很大，所以花了很大篇幅来介绍。
- ❑ 最后在案例中介绍了一个实用的文档对象——document 对象，及其属性、方法和事件。

在第 15 章中，将介绍几种常见的帮助页面设计人员工作的得力助手——可视化编辑软件。设计人员利用这些工具，可以实现更快的工作效率和更好的设计效果。

第 15 章　了解制作页面的工具

制作页面最基本的工具就是记事本，如果设计者愿意，完全可以用记事本做出任何页面。事实上，很多人也是这样做的。但是为了提高设计者的工作效率，出现了很多编辑页面的软件，闻名遐迩且运用最为广泛的是 Adobe 公司的一系列产品，本章将介绍这些主流的网页编辑工具。本章的主要知识点如下。

- 使用 Dreamweaver 工具实现页面开发。
- 了解 Photoshop。
- 了解 Flash 视频文件。
- 了解 Flash 工具和它的一些使用。

15.1　可视化编辑页面工具：Dreamweaver

Dreamweaver 是美国 MACROMEDIA 公司开发的集网页制作和管理网站于一体的所见即所得网页编辑器。强大全面的功能使之受到许多人的喜爱，利用它可以提高制作页面的工作效率。

Dreamweaver 软件的历史说来已久，在早年版本的 Dreamweaver 中，制作页面虽然已经是基于 CSS 样式表的布局，但是考虑到一些老用户，而且几年前的设计页面思路并没有如今天一样从“HTML 制作框架、CSS 样式表来修饰，JavaScript 来发挥页面的动态效果”这样的角度出发，所以多少还是有照顾早期表格框架来布局页面的思路。

而在最近发行的 Dreamweaver CS3.0 版本中，带来了很大的改变。最明显的改变是设计页面的方式是以 DIV+CSS 的思路为出发点，无疑新版本的 Dreamweaver CS3.0 将会给设计者带来更多的易操作性。

15.1.1　了解 Dreamweaver 的界面

当使用者第一次打开 Dreamweaver 时，软件会要求使用者选择“创建一个新页面”。这时软件会给出多种不同类型页面的选择项，如 PHP、HTML 和 XML 等。事实上，对于初学者来说，设计一个完整的页面，可以从选择 HTML 开始。一个基本的 Dreamweaver 界面样式如图 15.1 所示。

大体上整个 Dreamweaver 软件主要可以分为菜单栏、工具栏、预览区、代码区、属性修改和侧栏。从图 15.1 中可以很容易地注意到最显眼的两个部分，即预览区和代码区。Dreamweaver 最大的特色就在于当设计者在代码区域内写入完整正确的 HTML 代码时，在预览区中将可以看到页面的预览样式。

在编写代码时，Dreamweaver 会选择不同的颜色来表示不同的功能作用的代码，而且软件中预设了几乎所有的 HTML 标签和 CSS 样式表的属性。所以，编写代码时会自动生成可供选择的属性，这对设计者提高编写速度有很大的帮助。



图 15.1　Dreamweaver CS3.0 界面

虽然在预览区中能看到页面的预览效果，但由于事实上页面在不同的浏览器中效果都并非一样，所以有时设计者还是不得不在浏览器中查看最终的效果。

当选中页面中某一个页面对象，如文本、图像等时，在属性修改编辑栏中，图 15.1 中最下部分，虽然这里可以方便地通过预设的属性项目进行对页面内容编辑，但并不是一个制作页面的好方法，最好的方法依然是通过自己手动编写 CSS 样式表来修饰页面的内容。而通过属性面板修改页面对象时，会很容易出现难以意料的繁琐代码，这会让整个页面源码显得很糟糕。

界面的侧栏中有一些属性修改的面板，这个位置中有一个非常显眼的部分：CSS 样式表属性编辑面板，如图 15.1 右侧所示。在这个窗口中，设计者可以方便地制定样式表，如图 15.2 所示。

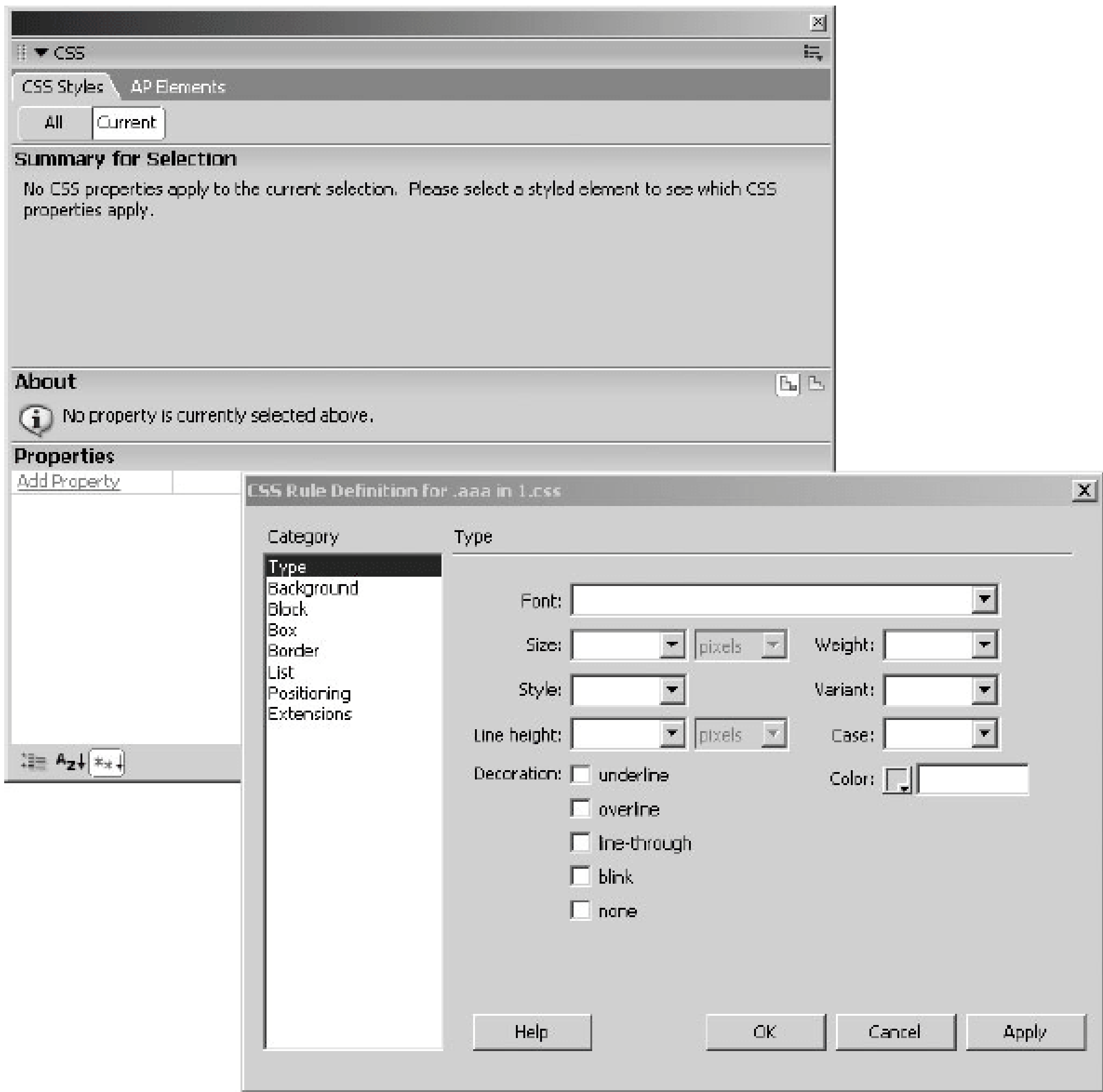


图 15.2　CSS 样式表编辑器

在这个编辑器中，当需要创建样式表时，会弹出窗口选项，如图 15.2 中下方窗口，使用者可以在这里方便地选择需要修改的属性。这令开发页面的初学者也可以很容易地上手操作。



说明：Dreamweaver 软件的使用不是本书的重点，故只做入门介绍。有兴趣的读者可以查找一些关于 Dreamweaver 的书籍参阅。

15.1.2　Dreamweaver 的菜单

和大部分平时所见的软件一样，就像汽车总是有些基本的特征，如 4 个轮子、车头车尾总是不会变的。如图 15.1 中的 Dreamweaver 界面，顶部是菜单栏，这在大部分软件中都是这样。Dreamweaver 中主要的菜单介绍如下。

- File：文件，在该菜单下可以进行打开、新建、保存这些基本的操作，同时也有一些导入、导出其他素材等功能。
- Edit：编辑，在该菜单下可以编辑对页面的操作动作，如一些基本的复制、删除、粘贴等。
- View：查看，在该菜单下可以对页面的显示方式进行编辑，如尺寸、网格线这样的功能来编辑页面布局。
- Insert：插入，在该菜单中可以在页面中插入不同的页面文件，如图像、表格、超链接或表单等。
- Modify：修改，该菜单下针对页面中不同的对象进行修改，如修改表格的行列属性、CSS 样式表的属性等。
- Text：文本，该菜单下可以对文本进行专门的编辑，如修改文本的字体大小、字体样式等。
- Commands：命令，在该菜单下可以预先设置好一组命令，当对不同对象使用相同操作时，可以无须再次重复编辑。
- Site：站点，用来链接整理服务器。
- Window：视窗，控制在界面中打开不同的窗口。
- Help：帮助，这里可以通过连接互联网查找一些关于 Dreamweaver 的使用资料。

15.2　神奇的“美工笔”：Photoshop

对于前端页面开发者来说，当拥有布局页面能力的同时，最好能拥有一些编辑图像的能力。在诸多图像编辑的软件中，Photoshop 的名声响彻业界，设计者常用的一种方法是在 Photoshop 中做好页面的设计图，然后放到 Dreamweaver 中进行编辑。本节将介绍如何使用 Photoshop 的基本功能。

15.2.1　了解 Photoshop 的界面

由于 Photoshop 和 Dreamweaver 现在同属于 Adobe 公司。所以它们在界面上样式十分类似，如图 15.3 所示。



图 15.3　Photoshop 界面

从界面上可以看出，Photoshop 基本上也是具备菜单栏、工具栏、工作区域，即图中的图像窗口和侧栏。Photoshop 相对于 Dreamweaver 而言，并没有特别多的选项。当然，这不是说 Photoshop 的功能不够强大，实际上它可以做出任何你能想象的图像。这里简单介绍一下 Photoshop 的菜单。

- 文件：作用如 Dreamweaver 软件，实现一些基本的功能，如保存、打开图像、导入导出等功能。
- 编辑：同样具备 Dreamweaver 软件中菜单的编辑功能，只是作用于不同的对象，此外，还有专门属于 Photoshop 的编辑工具栏属性的一些功能。
- 图像：可以修改图像的大小、画布的大小等功能。
- 图层：可以使用于图层的一些编辑方法。图层是 Photoshop 中一个重要的功能。
- 选择：选取编辑对象。
- 滤镜：对所选对象进行效果编辑，这是 Photoshop 中十分有魅力的一项功能，它可以令设计者的图像展示出难以想象的奇特效果。
- 分析：对图像进行测量工作。
- 视图：可以选择如标尺、网格等特殊工具来划分图像中不同的位置。
- 窗口：选择在界面中显示或者隐藏部分功能的窗口。
- 帮助：可以通过互联网查找一些关于如何使用 Photoshop 的资料。

15.2.2　Photoshop 的实用技巧

使用 Photoshop 时，可以通过工具来完成大部分操作。而在 Photoshop 中，这些主要的工具栏放在整个软件界面的左侧，如图 15.4 所示。

Photoshop 提供的工具并不是很多，但它能够做出任何人们可以想象出来的图像。Photoshop 编辑创建图像完全可以认为是一门博大精深的艺术学科。如图 15.4 所示，其中 15 号位置的是画笔工具，它可以描绘出千姿百态的图像；8 号位置的是钢笔工具，可以建立任何形状的图像轮廓；19 号位置的是文字工具，可以在图像中添加文本。

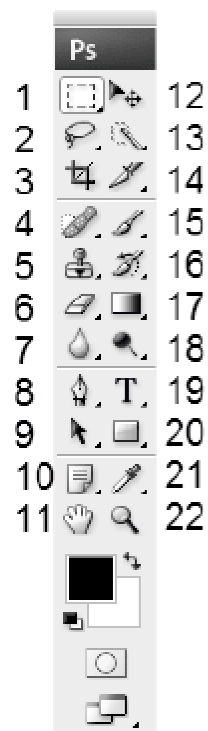


图 15.4　Photoshop 工具栏



说明：Photoshop 的内容已经超出了本书的范畴，这里只做一个简单的抛砖引玉，目的是帮助初学者有个简单的入门概念。有兴趣的读者不妨去查阅更多的关于 Photoshop 的书籍。

在很多时候，设计者制作网页时不可能每一个元素都是从零开始创新。例如，改变一幅图像的部分内容，显然不可能重头再去制作一幅基本差不多的图像。而比较好的方法是在原先图像的基础上进行修改。又如，当设计者只需要一个图像部分内容时，只要在素材上截取就可以了。



图 15.5　施工 1.png

通过图 15.4 工具栏中的工具，可以对图像进行一些基本的简单操作。这里介绍一种很实用的技巧——截取图像的技巧。下面先看如图 15.5 所示的一幅图像。

这是一张来自于互联网的图像，这里希望有这个“施工标志”的部分来为己所用，让它来点缀自己的页面。那么，该如何通过 Photoshop 来实现呢？

首先，在 Photoshop 中打开这张图像。为了截取这部分图像，可以使用几种工具。在图 15.4 中，1 号位置的工具是选择工具，它可以设置成矩形选框、椭圆形选框等。当选择好局部图像之后，可以通过剪切或者复制这部分图像，然后放在新的图层中，最后选择保存这个图层，如图 15.6 所示。



图 15.6　建立新选区的“施工标志”

此外，还有一种更直接的方法，使用工具栏中 3 号位置的截取工具，它可以直接将画布截取出来，如图 15.7 所示。



说明：画布和图像是两个概念，画布指图片的整体大小，图像指图片中内容部分的大小。



图 15.7　使用截取工具建立新图像



说明：如果最终得到的图像大小依然无法令设计者满意，可以使用 Ctrl+T 组合键来改变图像的大小。

这样截选出来的图像另存为适合的格式，如.jpg、.png 格式等，就可以放在网页中使用了。

15.3　网页中的动画：Flash

Flash 是一种文件，这里要提到的也是同属于 Adobe 公司的一款软件。当然，它的作用就是用来制作 Flash 文件的工具。但由于现今 Flash 文件的应用范围很广泛，所以 Flash 软件也可以说不仅是制作简单动画效果的 Flash 文件，同时也是可以制作出精美动态页面设计的软件。

15.3.1　认识 Flash 文件

Flash 最初给人的印象就是比 GIF 格式更漂亮一些的动画。Flash 目前是最为流行的一种矢量动画格式文件，其优点在于使用向量运算（Vector Graphics）的方式，产生出来的影片占用存储空间较小。Flash 文件的格式后缀名是.swf，如果需要在页面中加载 Flash 文件，则需要有特殊的播放器支持。

发展至今，Flash 所能做的已经不再是当初的一个小小的页面动画。其结合 ActionScript 语言的对象和流程控制，已经完全可以做到成为可以和用户互动的前端页面。甚至 Flash 也可以制作成为简单的游戏。

- 后缀名为.fla 的文件：如果制作一个 Flash 动画，只能通过 Flash 的源文件。源文件的后缀名

是.fla，在 Flash 软件中可以编辑这种格式的文件。由于浏览器并不支持这种格式，所以.fla 格式的文件是不能放入页面中使用的。

- 后缀名为.swf 的文件：而如果是希望放入到页面中的 Flash 文件，它的格式后缀名是.swf，这是将.fla 格式的文件经过压缩得到的一种小容量的 Flash 文件格式。

15.3.2　在页面中放入 Flash 文件

在页面中放入 Flash 文件不会太困难。从某个角度来说，它的原理和放入一张图像、一段 GIF 动画是一样的。所不同的是，如果在页面中放入 Flash 文件，用户却未必能够看到所希望的结果，原因在于 Flash 文件需要特定的播放器来加载文件，如 Flash Player。为此，可以通过一个简单的 JavaScript 程序来令系统下载，使浏览器支持 Flash 文件的插件，如下代码所示：

```
<script src="Scripts/AC_RunActiveContent.js" type="text/javascript"></script>
```

在页面源码<head>标签内声明引用了这样一个.js 外部文件，这个小程序的作用就是令浏览器下载支持 Flash 文件 swflash.cab 插件。同时，这个外部文件也可以防止在不同的浏览器中显示不同的结果。



说明：具体程序源码参考：资料光盘\第 15 章\script\AC_RunActiveContent.js。

在这段代码中声明了下载令浏览器支持 Flash 文件的小插件，这样设计者就可以在页面中加载 Flash 文件了。这里要通过一个<object>标签来将对象放入到页面中，具体如程序 15.1 所示，其是通过<object>标签将 Flash 放入到页面中的方法。

【本节示例参考：资料光盘\第 15 章\15-1　通过<object>标签将 Flash 放入到页面中.html】

【实例 15-1】通过<object>标签将 Flash 放入到页面中，其源码展示如下:

程序 15.1　通过<object>标签将Flash放入到页面中.html

```
01   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03   <html xmlns="http://www.w3.org/1999/xhtml">
04   <head>
05   <title>通过<object>标签将 Flash 放入到页面中</title>
06   <script src="Scripts/AC_RunActiveContent.js" type="text/javascript"></script>
07   </head>
08   <body>
09   <h1>这里是一个 Flash 文件</h1>
10   <script type="text/javascript">
11      AC_FL_RunContent(
12   'codebase','http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version
13   =9,0,28,0','width','550','height','450','title','Endf','src','end','quality','high','pluginspage','http://ww
14   w.adobe.com/shockwave/download/download.cgi?P1_Prod_Version=ShockwaveFlash','movi
15   e','end' );
16   //end AC code
```

```
17   </script>
18   <noscript>
19     <object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
20    codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#versio
21    =9,0,28,0" width="550" height="450" title="End">
22         <param name="movie" value="end.swf" />
23         <param name="quality" value="high" />
24         < embed src="end.swf" quality="high"
25   pluginspage="http://www.adobe.com/shockwave/download/download.cgi?
26   P1_Prod_Version=ShockwaveFlash" type="application/x-shockwave-flash"
27   width="550" height="450"></embed>
28     </object>
29     </noscript>
30     </body>
31    </html>
```

【运行程序】最终在页面中的效果如图 15.8 所示。



图 15.8　将 Flash 放入到页面中

【深入学习】代码第 19 行是<object>标签的起始，其中 clsid 是 class ID 的缩写。对于每个组件类，都需要分配一个唯一表示它的代码，就是 ID。为了避免冲突，微软使用 guid 作为 clsid。guid 的函数主要是根据当时的时间、机器地址等信息而动态产生，这样理论上可保证全球唯一。所以那一串数字就是 ID 号。

<object>标签中可以加入 width 或者 height 属性来修改播放窗口的大小。当然，如果使用样式表设计者可以做到更多不同样式的窗体表现。第 20 行代码中 codebase 是指一个程序集的位置，通过设置 href 特性告知运行库按照给定的 URL 来查找程序集。这里是引用到 Macromedia 的一个页面，它的功能是加载了 Flash 播放器。

从第 22 行代码开始，<param>标签来确定对象的更多细节，<param>标签中关联两个属性，分别是 name 属性和 value 属性。name 属性用来区别不同的<param>标签，如代码写为“<param name="movie">”和“<param name="balance">”，这样就是两个不同的<param>标签。name 属性关联不同的 value 值，这样就分开了不同的作用，如程序 15.1 中第 22 行和第 23 行所示。第 22 行代码表明了引用的 Flash 文件的位置。

第 23 行代码表明它的画面质量是“高”级别的。当设置为不同的 name 属性时，可以触发不同的作用，如表 15.1 所示。

表 15.1　<param>中name和value的设置方法

name	value	作　用
movie	some.swf	引用一个Flash文件
playcount	n	表示循环播放n次
autostart	1或者0	1表示页面打开自动播放，0表示需要单击播放
clicktoplay	1或者0	1表示允许播放和暂停动画，0表示禁用此项功能
displaysize	0、1或者2	控制播放画面：x=0，原始大小；x=1，一半大小；x=2，两倍大小
enablefullscreen controls	1或者0	控制切换全屏：x=1，允许切换为全屏；x=0，禁用此功能
enabled	1或者0	1表示可以人为控制播放器，0则表示不允许
fullScreen	1或者0	1表示全屏，0则不是
uiMode	full mini none invisible	播放器显示模式，full显示全部；mini最简化；none不显示播放控制，只显示视频窗口；invisible全部不显示
defaultFrame	1或者0	显示默认框架
rate	允许小数	播放速率控制，1为正常，1.0+则变快，反之变慢
volume	百分比	正常声音大小是100%
mute	1或者0	1表示正常，0表示静音

说明：从第 24～27 行代码中，在<embed>标签中的内容实际上的作用和前面的<object>部分是一样的，只是为了不支持<object>标签而存在的。现在<embed>标签已经渐渐地被淘汰了。

15.3.3　制作 Flash 的软件

之前已经见识了同属于 Adobe 公司的 Dreamweaver 软件和 Photoshop 软件，前者用来制作页面，后者用来制作静帧图像。而本节介绍的同样是 Adobe 公司的 Flash8 软件，如图 15.9 所示，它是用来专门制作 Flash 的软件。

Flash8 看上去比 Dreamweaver 和 Photoshop 都要复杂。这是因为 Flash 文件制作时，牵涉“时间”这样的概念，动画是有时间长度的。所以 Flash8 中有属于自己特有的一个时间轴窗口，如图 15.9 中标注所示。其他部分与 Dreamweaver 和 Photoshop 都是比较相似的，如菜单栏、工具栏和一些常用的窗口排列在侧栏。

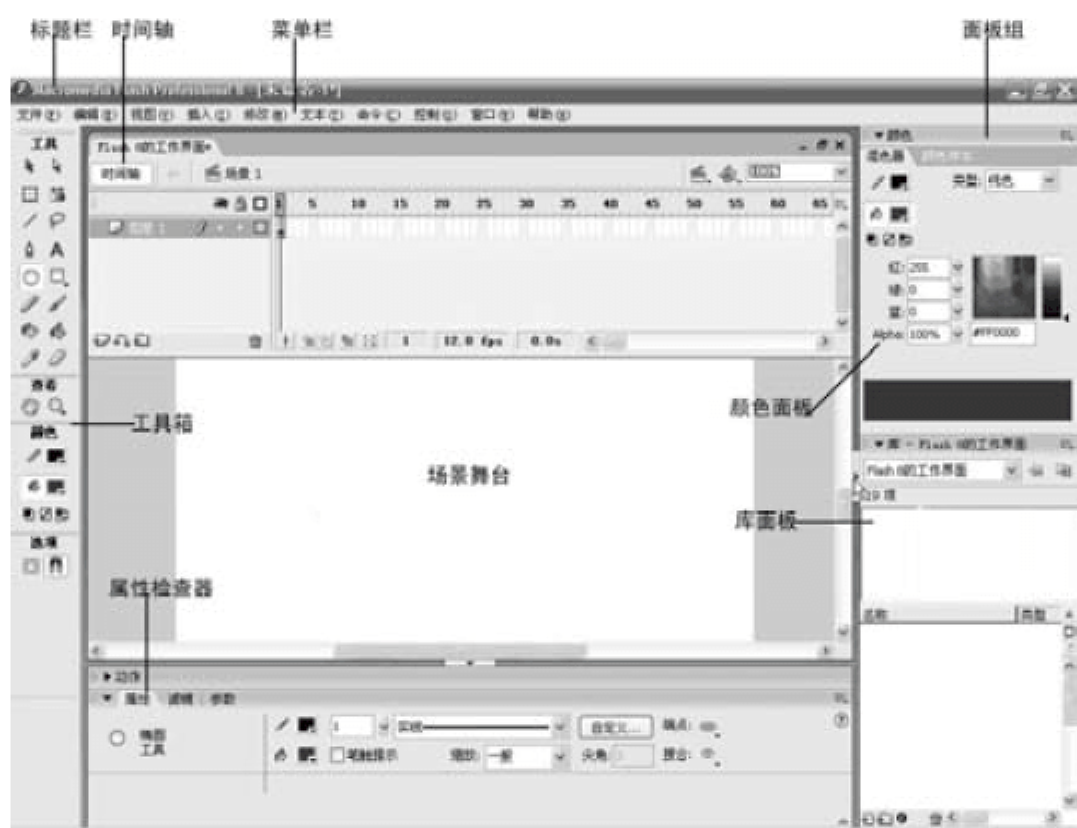


图 15.9　Flash8 软件界面

15.3.4　Flash8 的菜单

由于 Flash8 也属于 Adobe 公司，Flash8 的菜单分类也多少类似于 Dreamweaver 和 Photoshop。下面对它们进行介绍。

- 文件：进行打开、保存文件等基本操作。
- 编辑：对 Flash 构成的图像进行编辑的操作。
- 视图：使用一些测量方式来定位编辑对象的位置。
- 插入：插入时间轴，这是 Flash8 特有的一项功能。时间轴是控制动画的一项重要的工作区域。
- 修改：针对于 Flash 中的元素进行修改，如位图、元件、时间轴等。
- 文本：编辑文本的字体等属性。
- 命令：通过调用预先设置好的命令进行快速操作。
- 控制：控制 Flash 文件最终的预览效果。
- 窗口：选择出现在软件界面中的工具窗口。
- 帮助：可以通过互联网查找关于 Flash8 的资料。

其实无论 Dreamweaver、Photoshop，还是 Flash8，它们基本的操作对象大部分都属于图像、文本，只是在不同的环境中需要不同的方法来操作对象，所以这 3 种软件在大部分的功能对象上都具有很大的类似性。

15.3.5　Flash8 的主要功能

Flash8 的工具栏界面和 Photoshop 工具栏大同小异。当然，由于 Flash 文件主要是由矢量图形构成的，所以这些工具在使用时也会有不同的感觉，如图 15.10 所示，这些工具也能完成矢量图形制作。但是 Flash 软件主要的功能并非用于设计静帧图像，而是一方面用于制作矢量图形的动画，另一方面可实现 Flash 组件的交互来制作页面，这需要了解 ActionScript 语言。

说明：制作 Flash 不在本书讨论的范围之内，有兴趣的读者可以参考相关类别的书籍。



图 15.10　Flash8 工具栏

15.3.6　Flash 的常用交互技巧

目前，主流网站中都具备给页面添加视频的功能，这就是通过 Flash 组件来实现的。在了解如何添加的方式之前，要明白什么是视频文件。平时生活中，人们看的 DVD、摄像机拍摄的视频片段，或者是数码相机拍摄的短片，又或者是网络上下载的影片等。这些视频绝大多数是以.avi、.rmvb、.mov 等为后缀名的文件。

但是，这些文件容量较大，实在不适合加载到页面中使用，直到 FLV 文件出现。FLV 视频格式占有率低、视频质量还算凑合，但体积相当小，这些特点尤其适合被加载在网络中。所以 FLV 视频文件是目前网络视频的主流文件。

FLV 就是随着 Flash 软件发展而来的视频格式，目前被众多新一代视频分享网站所采用。它是目前增长最快、最为广泛的视频传播格式。它是在 Sorenson 公司的压缩算法的基础上开发出来的。FLV 格式不仅可以轻松地导入 Flash 中，并且能起到保护版权的作用，而且可以不通过本地的微软或者 Real 播放器播放视频。

接下来通过 Flash8 了解如何实现在页面中添加视频文件。打开 Flash8，新建一个 Flash 文档，如图 15.11 所示。

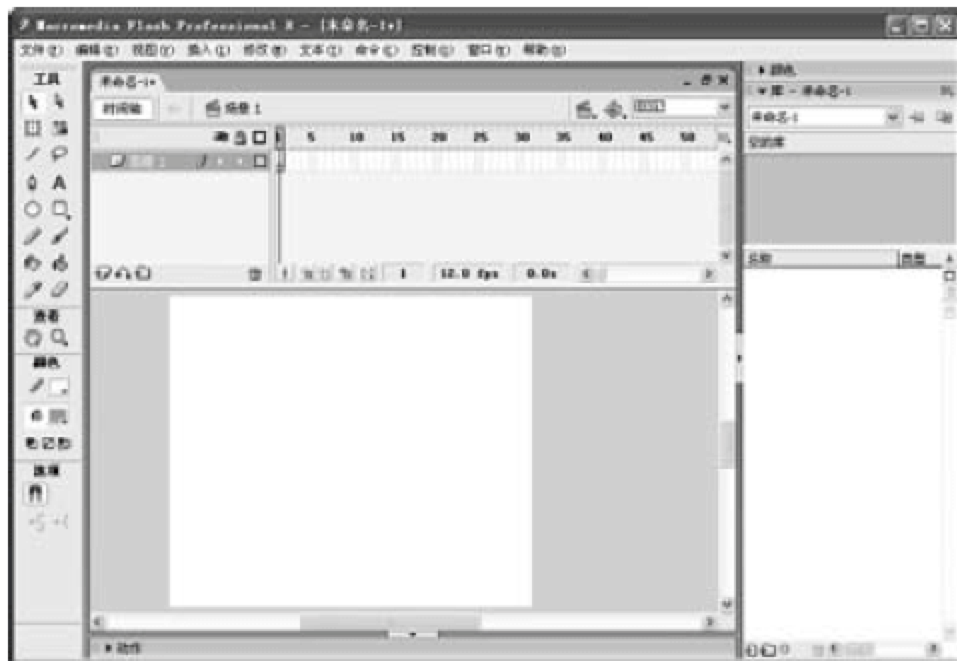


图 15.11　打开 Flash8 的界面

在“文件”菜单下新建一个大小为 400*300 的图层。这里可以在新建好的图层中将鼠标放在编辑视窗中右击，在弹出的快捷菜单中选择“文档属性”命令修改图层大小。

在“窗口”菜单下打开“Flash 组件”窗口，或者通过快捷键 Ctrl+F7 打开组件窗口，然后将“组件”中的 FLVPlayback 拖入编辑窗口，这是一个播放器的功能，如图 15.12 所示。

接着在播放器中放入准备好的 FLV 视频文件，先选中编辑窗口，在属性面板中打开“参数”选项卡，如图 15.13 所示。

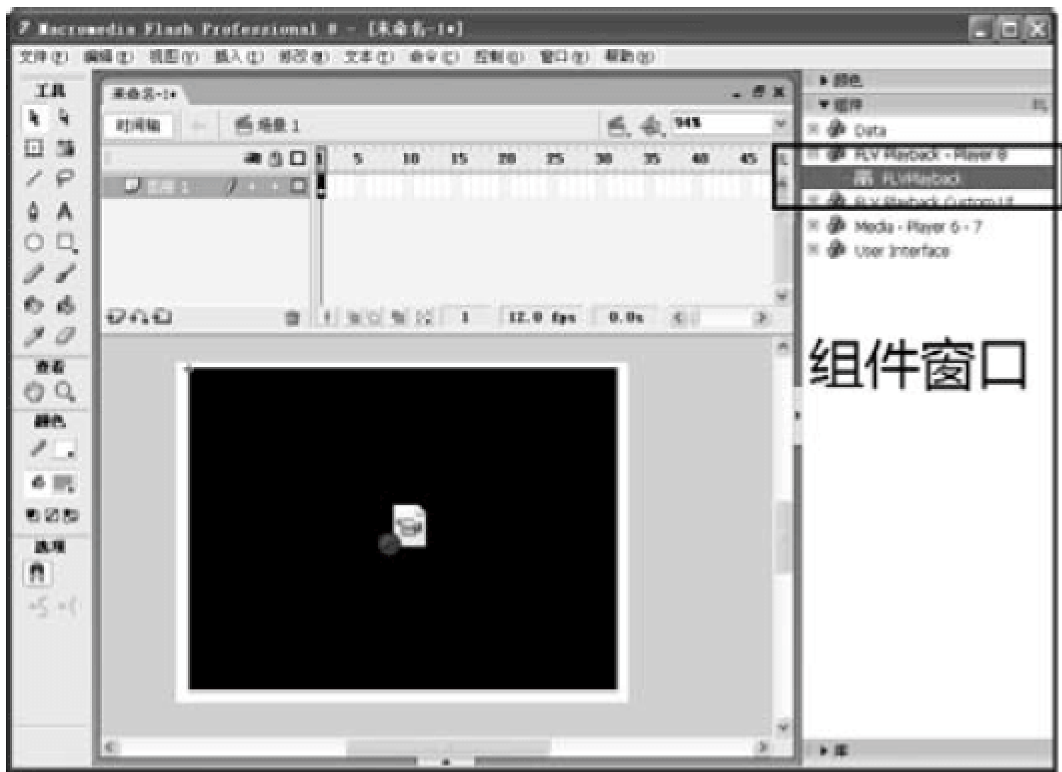


图 15.12　放入 FLV 播放器

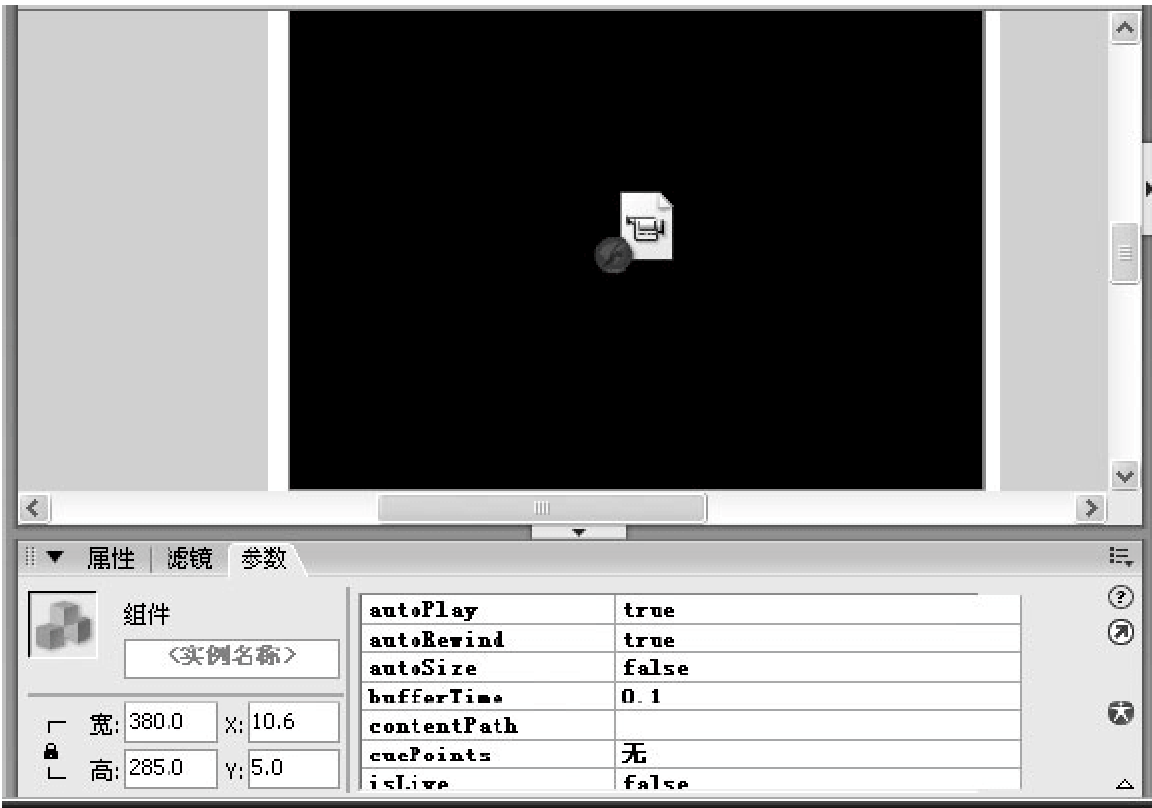


图 15.13　播放器的参数设置

说明：在属性面板中还可以修改播放器窗口的大小等属性。

此时，需要设置属性面板中的“参数”数值，图 15.13 中的下面部分。选择 contentPath 选项，表示添加 FLV 视频文件的路径。这里放入事先准备好的文件 because of you.flv，接着在 skin 选项中选择添加播放控制按钮的样式。最后选择“文件”|“导出”|“导出影片”命令，即保存为.swf 后缀的 Flash。把这个 Flash 视频加载到页面中，这样就完成了一个页面中加载视频的过程。

15.4　案例：使用 Dreamweaver 制作页面

本节将通过一个实例来展示如何合理使用 Dreamweaver 创建页面。Dreamweaver 中包含了大部分主流布局的框架，其相当杰出的一点是它的模板设计，如图 15.14 所示。

（1）选择 File|New 命令，弹出模板菜单选择界面。从模板菜单选项中可以看到有很多不同类型的布局模板。这里选择的是一个普通的左右分栏和上下栏的布局结构。当创建好之后，可以在可视编辑窗口中看到已经生成的代码和页面结构，如图 15.15 所示。

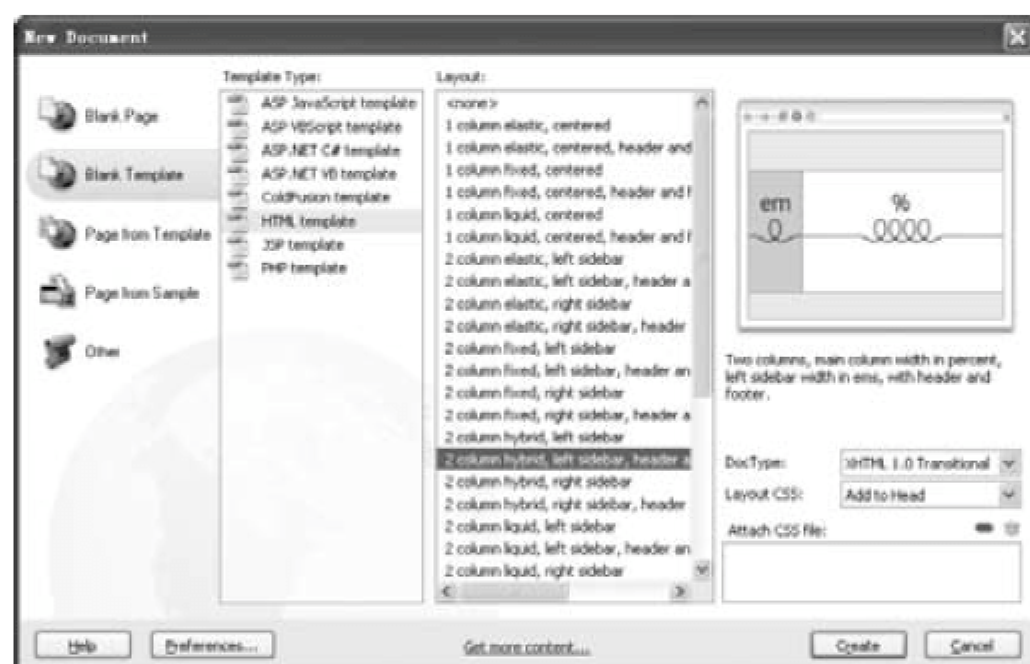


图 15.14　Dreamweaver 的选择布局界面



图 15.15　创建好的页面布局

（2）如图 15.15 所示，软件界面中，上面部分是代码生成的地方，而下半部分是可预览的最终效果，按下 F12 键，可以查看在浏览器中的预览效果，如图 15.16 所示。



图 15.16　在浏览器中的预览结果

当然，只是这样简单地使用 Dreamweaver 就希望能够成为一个尖端页面开发高手显然是不够的。起码当软件为使用者快速补全代码的同时，作为使用者，需要能够很好地理解 Dreamweaver 生成的代码。否则，非但 Dreamweaver 不能帮到你，相反很容易令页面充满大量的垃圾代码。

15.5　小　　结

本章简单地介绍了目前主流商业化网站创建的“利器”，之所以有这样的概念，就是软件始终只是一种辅助性工具，要做好前端页面的设计工作。HTML+CSS 的基础知识是非常重要的，只有在理解

的基础上再去创新，才能创造出优秀的页面，而不是一旦脱离了辅助工具就变得手足无措。本章的主要知识点有：

- ❑ 了解 Dreamweaver 是一种可视化的页面开发工具。只有在了解 HTML 语言的基础上，使用 Dreamweaver 才能发挥作用；否则，反而会给设计者带来更多的麻烦。
- ❑ 了解 Photoshop 是用来制作图像的工具。通过 Photoshop 软件，可以完成页面中图像的设计、编辑工作。
- ❑ 了解 Flash 视频文件的特性以及 Flash 软件是开发这种文件的工具。Flash 软件可以将.fla 格式的文件导出为.swf 文件，并将其放入页面中。

在之后的章节中，会通过几个综合性的例子，纵观一个整体的页面开发是如何实现的。

第 4 篇　页面实战篇

第 16 章　综合实例一：制作主流网站界面

第 17 章　综合实例二：设计复杂页面

第 18 章　综合实例三：制作英文网站

第 19 章　综合实例四：使用 Dreamweaver 制作中文网站



第 16 章　综合实例一：制作主流网站界面

在本章中，将综合一些基本的知识技能，详细介绍一个页面是如何从构思到规划，最终成型展现在互联网上，如图 16.1 所示。这是目前大部分网站所运用的一种页面布局。布局中分为头部、底部和中间主要部分。其中，中间的部分又通常被拆分成几列，用来放置不同的页面内容。



图 16.1　常见的页面布局

本章的主要知识点如下。

- ❑ 构思页面的布局。
- ❑ 制作页面的顺序。
- ❑ 理解制作页面的基本思路。

【本节示例参考：资料光盘\第 16 章\案例一.html】

16.1　构思基础的布局

从前面的知识中已经了解到如何设计一个页面。首先，设计师心中需要明确摆放哪些内容，接着就是构思如何去布局这个页面。



技巧：通常为了便于工作，在 Photoshop 中设计出大概的草图，就像图 16.1 那样。在这个草图中，设计师可以大致将页面的布局规划出来，如图 16.2 所示。



图 16.2 规划页面结构

可以把页面分割成这样的 4 部分。页面的顶部 Header、底部 Footer、中间部分 Main，其中 Main 又可以分成侧栏 Side 和主要部分 Main。所以，在编写代码时，它的结构看上去应该是这样的。程序 16.1 为初始页面的结构性代码。

【代码 16-1】初始页面的结构性代码，其源码展示如下：

程序 16.1 主页

```
01   <body>
02      <div id="container">                    //页面层容器
03            <div id="header">                 //页面头部
04            </div>
05               <div id="Main">                //页面主体
06                     <div id="Side">          //侧边栏
07                     </div>
08                 </div>
09          <div id="Footer">                   //页面底部
10          </div>
11      </div>
12   </body>
```

【深入学习】代码中通过 div 定义了页面的几个组成部分：头部、主题、边栏和底部。它们都包含在第 2 行定义的 container 层中。

这里已经设计好了一层层的框模型。接下来就是针对于每一块区域，设计好针对于每一对象的样式表。

16.2 设计基础模块的样式表

针对不同的结构部分，设计好相对应的 CSS 样式表，如程序 16.2 所示。

【代码 16-2】基本结构所对应的样式表，其源码展示如下：

程序 16.2　CSS设计

```
01   <style type=text/css>
02          body {
03            font:12px 微软雅黑;                      //基本信息
04             margin:0px;
05             text-align:center;                     //使页面文本居中
06             background:#FFF;
07             }
08   #container {
09                   width:100%;                      //页面层容器
10                    }
11          #Header {
12                 width:800px;                       //页面头部
13                  margin:0 auto;                    //使#Header 部分自动页面居中
14                  height:100px;
15                  background:#FFCC99;
16                  }
17           #Main {
18                     width:800px;                   //页面主体
19                     margin:0 auto;                 //使#Main 部分自动页面居中
20                     height:400px;
21                     background:#CCFF00;
22                     }
23           #Side {
24               float: left;                         //侧栏
25              width: 20em;
26              background: red;
27              padding: 15px 0;                      //设置侧栏的空距
28                   }
29            #Footer {
30                  width:800px;                      //页面底部
31                  margin:0 auto;                    //使#Footer 部分自动页面居中
32                  height:50px;
33                  background:#00FFFF;
34                  }
35   </style>
```



说明：这里为了令读者能够容易辨识不同的模块，所以每一个 CSS 样式表中都添加了背景颜色。事实上，在最终的页面中并不需要这样做。

【深入学习】上述代码定义的是前面设计好的页面，其中第 3～6 行是整个页面的样式，包括字体、背景色等。代码第 23～28 行是侧边栏的样式，包括它的宽度、背景色等。

【运行程序】它目前在浏览器中看起来如图 16.3 所示。



图 16.3　页面布局的结构效果

这样，一个基础的框架就出来了，接下来就需要细化工作，去逐一完成每一个框模型的对象。

16.3　完善网站的子模块

基于已经创建好的页面结构一点点地去细化完成网站的各个部分，如头部、底部、侧栏和页面的主体部分。只要按部就班，这些都不会很难。

16.3.1　网站的导航栏

希望最后出现的效果如图 16.4 所示。



图 16.4　导航栏的效果

大部分导航栏都是通过列表项和行内模块展现出来的，导航栏是常用的页面元素之一，这个导航栏就是比较典型的一种。如程序 16.3 中导航栏的制作方法。

【代码 16-3】制作导航栏，其源码展示如下：

程序 16.3　导航栏

```
01  <div id="tabs">
02    <ul>
03    <li><a href="#" title="菜单 1"><span>菜单 1</span></a></li>
04    <li><a href="#" title="菜单 2"><span>菜单 2</span></a></li>
05    <li><a href="#" title="菜单 3"><span>菜单 3</span></a></li>
06    <li><a href="#" title="菜单 4"><span>菜单 4</span></a></li>
07    <li><a href="#" title="菜单 5"><span>花样年华</span></a></li>
08    <li><a href="#" title="菜单 6"><span>博客</span></a></li>
09    <li><a href="#" title="菜单 7"><span>联系我们</span></a></li>
```

```
10      </ul>
11   </div>
```

【深入学习】代码第 3～9 行通过<li>罗列出了菜单项，当用户单击每个菜单项时，通过 href 属性指定的位置可以导航到目的地。

那么，针对于这个导航栏，它的 CSS 样式表的写法如程序 16.4 所示。

【代码 16-4】制作导航栏的样式表，其源码展示如下：

程序 16.4　导航栏的样式表

```
01        #tabs {   position:relative;                        //#tabs 层的样式定义
02                  float:right;
03                  width:100%;
04                  font-size:93%;
05                  border-bottom:1px solid #F45551;          //设置边框的样式
06                  line-height:normal;
07                  }
08        #tabs ul {
09            margin:0;
10            padding:10px 10px 0 50px;                      //设置空距样式来修饰导航栏
11            list-style:none;                               //取消项目列表符号
12                    }
13        #tabs li { display:inline;                         //使列表项横向排列
14                        margin:0;
15                        padding:0;
16                    }
17        #tabs a {   float:left;
18                    background:url(tableft9.gif) no-repeat left top;    //给条目添加背景图像
19                    margin:0;
20                    padding:0 0 0 4px;
21                    text-decoration:none;                  //取消列表项链接下划线
22                    }
23        #tabs a span {float:left;
24                        display:block;
25                        background:url("tabright9.gif") no-repeat right top;
26                        padding:5px 15px 4px 6px;          //使用空距修饰链接效果
27                        color:#FFF;
28                           }
29        // IE5 hack                                        //设置 Hack 以适应 IE5 浏览器
30        #tabs a:hover span {
31           color:#FFF;
32           }
33        #tabs a:hover {
34           background-position:0% -42px;
35           }
36        #tabs a:hover span {
```

```
37            background-position:100% -42px;
38            }
```

【深入学习】第 1～7 行定义了 tabs 层的样式，涉及字体、宽度等。第 13～16 行的样式定义特别关键，其中第 13 行表示将列表项横向排列。



说明：第 29 行之后是针对于 IE 5 之前版本的 Hack。

16.3.2　页面的侧栏

页面的侧栏是一个浮动层，它的中间可以放入任何页面对象。通常，一些简单的条目适合放在这里，作为目录使用。而页面的主体部分，放入一些具体对象的文本描述，最好再添加一些图像，令这个页面变得更加生动。这个例子中，希望最终的样子如图 16.5 所示。

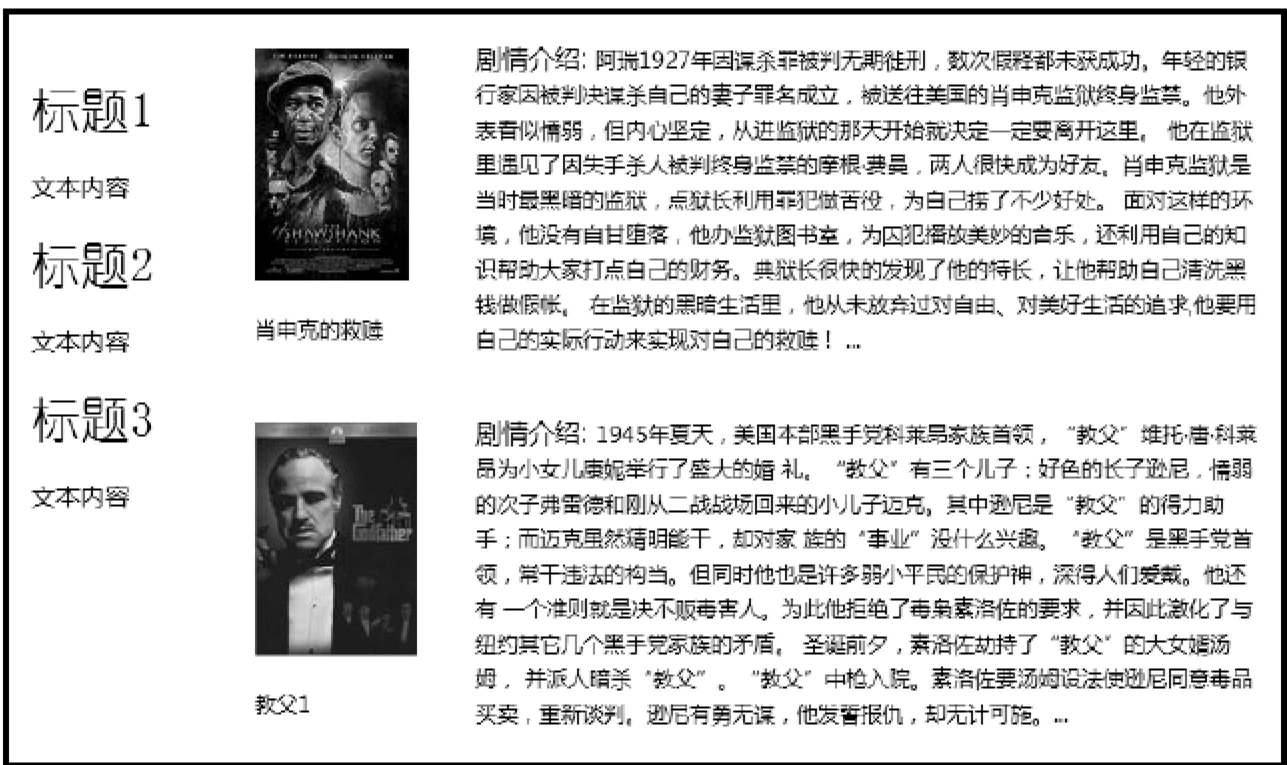


图 16.5　页面的主体

那么，这里需要更细化地去解决页面的结构性问题。左侧的浮动层不难，只要解决好文本的排列即可。而右侧主体部分，用了表格来划分图像和文本的内容。具体的写法如程序 16.5 细化页面中间结构。

【代码 16-5】页面中间部分的细化结构，其源码展示如下：

程序 16.5　细化页面中间结构

```
01      <div id="Main">
02          <div id="Side">
03            <div class="sidetext">
04              <h2>标题 1</h2>
05              <p>文本内容
06              <h2>标题 2</h2>
07              <p>文本内容
08              <h2>标题 3</h2>
09              <p>文本内容
10            </div>
11          </div>
```

页面侧栏，在页面的左侧

```
12              <div>
13                <table>
14                  <tr>
15                    <td id="film"><img src="图片/1.jpg" ><p>肖申克的救赎
16                    <td>剧情介绍: <span id="film"> 阿瑞 1927 年因谋杀罪被判无期徒刑…
17                  </tr>
18                  <tr>
19                    <td id="film"><img src="图片/2.jpg"><p>教父 1
20                    <td>剧情介绍: <span id="film"> 1945 年夏天，…
21                  </tr>
22                  <tr>
23                   <td id="film"><img src="图片/3.jpg"><p>教父 2
24                   <td>剧情介绍: <span id="film"> 在西西里，少年时代的维托为报父仇，…
25                  </tr>
26                  <tr>
27                   <td id="film"><img src="图片/4.jpg"><p>教父 2
28                   <td>剧情介绍: <span id="film"> 在警长戈登和地区检察官哈维·丹特的协助下，…
29                </table>
30              </div>
31      </div>
```

【深入学习】第 13～31 行定义了一个 table 表格，<tr>是表格的行，<td>是表格的列。

当基本的结构完成之后，需要设计对应于这个结构的样式表，这需要经过不断反复地查看效果，才能达到美观。如程序 16.6 编写的修饰页面的样式表。

【代码 16-6】修饰页面主体的样式表，其源码展示如下：

程序 16.6　对应于这个结构的样式表

```
01  .sidetext {padding:10;              //设置侧栏中文本的样式
02            color:black;
03              }
04  h2 {font:2em 幼圆;
05      }
06  td {color:navy;                      //设置表格中文本的样式
07        padding:15px;
08        height:200px;
09        border:1px solid white;
10        }
11  #film {font:.8em;                    //设置 film 层文本样式
12          color:black;
13          }
```

【深入学习】上述代码都是定义的样式，其中第 6～10 行是对前面表格中的文本设置样式，涉及文本与表格边线之间的距离和边线的颜色等。

最后一部分是页面的底部。页面的底部也是一个单独的层，它的代码十分简单：

```
<hr width="700">
<div id="Footer"><p>&copy; 2008-10-23
</div>
```



说明：<hr>表示的是“水平线”。

16.4　最 终 页 面

最终，将所有的代码放在一个文档中，可以看到最终的页面效果，如图 16.6 所示。



图 16.6　最终页面效果

16.5　小　　结

本章主要介绍了一个页面从构思到成型的过程。相同页面不同代码的例子有很多，这里给出了最易于理解的一种基础的做法，即便今后遇到更复杂的页面，其基于的原理都是一样的，都是建立于这样的一种制作思路。即将页面结构和表现分离开，针对不同的页面对象去进行编辑。在第 17 章中，将基于这样的结构页面，实现更多更细致的页面修饰，令页面更丰富。

第 17 章　综合实例二：设计复杂页面

本章将在第 16 章内容的基础上，将页面拆分得更细致化，并且在布局完成的页面基础上，熟悉如何去修饰页面、拓展页面的功能。学习了本章内容，掌握了这些千篇一律的操作技巧后，可以说足以应付市面上大部分主流的页面。本章的主要知识点如下。

- 构思页面的布局。
- 按照结构顺序制作页面。
- 如何设计对应于每个模块的样式表。
- 理解制作页面的整体思路。

17.1　页面的框架布局

在设计一个页面时，首先是内容的定位，其次是依据页面内容排版页面布局，以配合需要放入页面中的内容。当设计好初级的页面布局后，如果页面内容需要，则可以进一步去细分页面布局。当页面布局完成之后，最后可以在相应的位置按照预先的设计放入相应的页面修饰。

设计页面的方式有很多种，重要的是要保证页面源码的可读性、可扩展性和良好的兼容性。基于这个思路，本章同样是使用相同的方案去制作更复杂的页面。

【本章示例参考：资料光盘\第 17 章\案例二.html】

17.1.1　定位页面的内容

假如一个具备大量信息的基本门户页面需要较多有框模型的层，那么如何去处理好一层层的叠加，就是比较重要的一项工作。本章的实例是关于“页面设计的一个主题”，如图 17.1 所示。

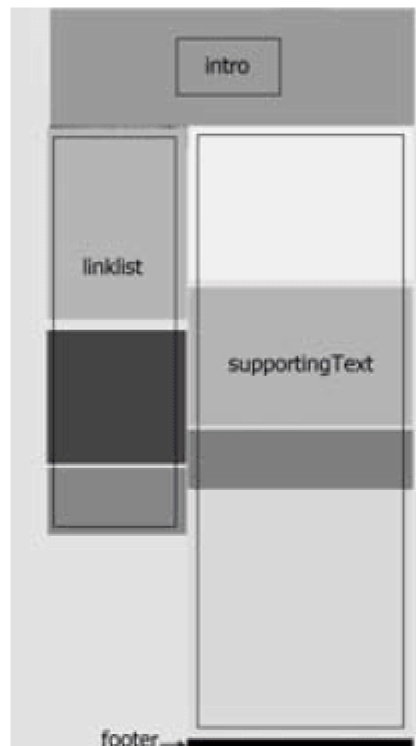


图 17.1　页面的初级布局

可以看出，首先页面的初级布局是一个分为 4 部分的框架，分别是 intro、linklist、supportingText 和 footer。其次，在大框架之下，也已经拆分出很多不同颜色块的框模型。



说明：这是为了针对不同的页面内容，它们各自属于不同的上级框模型，即初级布局中的 4 个框模型。

17.1.2　页面初级布局的代码

基于这样的框架，它的代码如程序 17.1 所示。

【代码 17-1】*页面初级布局的代码，其源码展示如下：*

程序 17.1　页面框架

```
01   <body>
02         <div id="container">                 //页面层容器
03               <div id="intro">               //页面的头部
04               </div>
05               <div id="supportingText">      //页面的右侧栏
06                     <div id="footer">
07                     </div>
08               </div>
09               <div id="linkList">            //页面的左侧栏
10               </div>
11         </div>
12   </body>
```



说明：footer 框模型部分是嵌套在 supportingText 框模型中，这样的设计是为了不希望页面页脚撑满页面的整个宽度，而是为了令它只是存在于页面主体的右侧中。如果设计者不希望这么做，完全可以把它独立出来。

【深入学习】代码第 2～11 行通过 div 将页面分成了几部分，包括头部、左侧栏和右侧栏。这 3 部分都放在一个名为 container 的 div 层中。

基于页面的结构性的代码，紧接着需要做好的便是如何通过 CSS 样式表精确控制整个页面的布局，如程序 17.2 所示为页面初级布局所运用的样式表。

【代码 17-2】*页面初级布局所运用的样式表，其源码展示如下：*

程序 17.2　页面框架的样式

```
01   body { font : 12px/17px  微软雅黑;                    //设定页面的基本样式
02          color : #000;
03          background : #dae8bd url(bg_lines.gif) no-repeat top 0%;
```

```
04          margin : auto auto;
05          padding : 0;
06          text-align : center;
07          }
08  #container { width : 762px;                //设置#container 层的宽度
09                padding : 0;
10                margin : auto;
11                text-align : left;
12                }
13  #intro { vertical-align : bottom;
14          }
15  #supportingText { padding : 0;
16                    margin : -3px 0 40px 0;  //设置#supportingText 层的位置
17                    background : #d6e6b6;
18                    border : 1px solid white; //设置边框的样式
19                    width : 469px;
20                    float : right;
21                    }
22  #footer { background : #a2c1b9;            //设置#container 层的宽度
23            border-top : 1px solid white;
24            padding : 0 0 2px 24px;          //设置空距的样式
25            }
26  #linklist { margin-top : 50px;             //设置#linklist 层的位置
27            clear : both;
28            padding : 20px 10px 10px 10px;
29            display : block;
30            }
```

【深入学习】在这段样式表中，可以看出它们对每一块的布局是如何定义的，body 定义了整个页面的初始背景、文本基本信息。container 则是定义了整个页面的主体部分，参照图 17.1 可以容易地分辨出 intro、supportText、footer 和 linklist 所定义的不同位置框架。

17.2　细化页面的局部

当页面的整体构架完成之后，接下来就是对局部的修饰和细化，在图 17.1 中已经可以看出在每一个初级框架下将要填入的内容，那么在本节中将一一来揭开这些模块神秘的面纱。

17.2.1　intro 部分

页面的头部在这里定义为 intro，依照这个例子中的需要，将主要使用图像来填充这一部分。当然，需要设计更详细的样式表来描述 intro 部分，如程序 17.3 所示为完善页面的 intro 部分。

【代码 17-3】完善 intro 部分的代码，其源码展示如下：

程序 17.3　intro部分

```
01  <div id="intro">
02          <div id="pageHeader">
03           <h1><span>更改样式表可以在这个位置替换页面的 banner
04           </span></h1></div>
05          <div id="quickSummary">
06              <p class="p1"><span>使用这个样式表可以任意更改页面右上角的图像</span></p>
07              <p class="p2"><span>设计复杂页面<a href="" >html</a> &<a
08                      href="">css</a></span></p>
09          </div>
10  …
11  </div>
```

【深入学习】上述代码在页面的头部添加了两个 div，第 7、8 行中虽然添加了两个链接，但都没有设置它的导航目的地。



说明：如果没有相应的样式表做出定位，通过上述结构性标签是无法窥探出页面的形态的。

为上述代码添加所对应的样式表，如程序 17.4 所示的 intro 部分的 CSS 样式表。

【代码 17-4】intro 部分的 CSS 样式表，其源码展示如下：

程序 17.4　intro部分的样式表

```
01  #pageHeader { padding : 0;                    //设置#pageHeader 层在页面中的表现
02                margin : 0;
03                height : 246px;                 //固定层的高度大小
04                border : 1px solid white;       //设置边框的样式
05                }
06  #pageHeader h1 { background : url(title_hdr.jpg) no-repeat top left;    //设置背景图像
07                   width : 760px;
08                   height : 176px;
09                   margin : 70px 0 0 0;         //设置不同边距大小
10                   padding : 0;
11                   }
12  #pageHeader h1 span { display : none;         //隐藏页面对象
13                        }
14  #pageHeader h2 { padding : 0;
15                   margin : 0;
16                   }
17  #pageHeader h2 span { display : none;         //隐藏页面对象
18                        }
19  #quickSummary p.p1 { width : 320px;
20                       height : 92px;
21                       position : absolute;     //定位其在页面的位置属性
```

```
22                  top : 1px;
23                  margin-left : 470px;
24                  padding : 0;                //设置空距为 0
25                  font-size : 11px;
26                  color : #fff;               //设置文本颜色
27                  font-family : 微软雅黑;
28                  background : url(bg_redbox.png) no-repeat;    //设置背景图像
29                  }
30  #quickSummary p.p1 span { display : none;
31                      }
32  #quickSummary p.p2 { display : block;       //隐藏页面对象
33                   position : absolute;       //绝对定位
34                   top : 53px;
35                   padding : 0 0 0 7px;
36                   font-size : 11px;
37                   text-align : right;
38                   color : #490909;
39                   font-family : 微软雅黑;
40                  clear : both;
41                  }
```

【深入学习】通过这样的样式表，就便于理解程序 17.3 了。程序 17.3 中第 2～4 行是 pageHeader 部分，在这段代码中，表明使用了背景图像来放置在页面头部的位置，这可以通过程序 17.4 的第 6～11 行看出来。之后为了令页面更具美观性，在页面的右上角放入一张修饰的图像 bg_redbox.png，如程序 17.4 中第 19～29 行所示。需要注意的是，pageHeader h1 span 样式表对象是被隐藏的，虽然这样看上去似乎没有什么用，但有时这样的做法便于配合 JavaScript 使用。



说明：quickSummary p.p2 对象定义了一个小小的标题栏在页面的左上角，这样做的目的是为了增加页面的美观度。所以从全局来说，intro 部分可以看成是由页面左上角的标题和右上角的图像以及页面的中间头部图像 3 部分组成。

【运行程序】浏览该页面，如图 17.2 所示。



图 17.2　页面的头部

17.2.2 页面的左侧部分

当设计好页面的头部后，接下来先处理页面的左侧，从图 17.1 中可以看出，页面的整个左侧栏都放在 linklist 对象中。下面来看一下左边侧栏的结构性代码，如程序 17.5 所示。

【代码 17-5】页面左侧部分结构性代码，其源码展示如下：

程序 17.5 页面左侧部分结构

```
01  <div id="linkList">
02      <div id="preamble">
03          <h3><span>设计法则</span></h3>
04      <p class="p1"><span>        
05      保持简单纯朴:
06          </span></p>
07      <p class="p2"><span>        
08      呼吸空间:
09              </span></p>
10              <p class="p3"><span>        
11              大就是美:
12              </span></p>
13  </div>
14  <div id="linkList2">
15                  <div id="lselect">
16                      <h3 class="select"><span>Select a Design:</span></h3>
17                      <ul>
18                          <li><a href="" title="AccessKey: a" accesskey="a">目录 1</a> —
19                          — <a href="" >name one</a></li>
20                          <li><a href="" title="AccessKey: b" accesskey="b">目录 2</a> —
21                          — <a href="" >name two</a></li>
22                          <li><a href="" title="AccessKey: c" accesskey="c">目录 3</a> —
23                          — <a href="">name three</a></li>
24                          <li><a href="" title="AccessKey: d" accesskey="d">目录 4</a> —
25                          — <a href="">name four</a></li>
26                          <li><a href="" title="AccessKey: e" accesskey="e">目录 5</a> —
27                          — <a href="">name five</a></li>
28                          <li><a href="" title="AccessKey:  f" accesskey="f">目录 6</a> —
29                          — <a href="">name six</a></li>
30                      </ul>
31                  </div>
32              <div id="larchives">
33                      <h3 class="archives"><span>Archives:</span></h3>
34                      <ul>
35                          <li><a href="" >下一页 &raquo;</a></li>
36                          <li><a href="" >内容简介</a></li>
37                      </ul>
38                  </div>
39  </div>
```

【深入学习】从这段代码中可知，linklist 框模型中包含了两部分，分别是 preamble 和 linklist2。而 linklist2 下是两个项目列表部分，分别是 lselect 和 larchives。所以在图 17.1 中，页面左侧由 3 个颜色块组成，这 3 个颜色块就是 preamble、lselect 和 larchives。它的样式表代码如程序 17.6 所示。

【代码 17-6】页面左侧布局的 CSS 样式表，其源码展示如下：

程序 17.6 页面左侧布局的样式表

```
01  #linklist { margin-top : 0px;                                //设置#linklist 在页面中的样式
02              padding:0px;
03              height:800px;
04              }
05  #linkList #linkList2 ul { padding : 20px 10px 10px 10px;    //设置空距的样式
06                            display : block;                   //设置为块级对象
07                            }
08  #linklist li { margin : 2px 0;
09               }
10  #preamble { padding : 0;                                     //设置#preamble 在页面中的样式
11                margin : -3px 0 0 0;
12                width : 288px;
13                float : left;
14                }
15  #preamble h3 { width : 288px;
16                   height : 47px;
17                   margin : 0;
18                   padding : 0;
19                   border-top : 1px solid white;
20                   }
21  #preamble h3 span { display : none;                          //隐藏对象
22                        }
23  #preamble p { font : 12px/15px 微软雅黑;
24                  padding : 0 0 0 3px;
25                  width : 288px;
26                  }
27  #preamble p.p1 { margin-top : 10px;
28                     }
29  #lselect, #larchives { width : 256px;
30                          clear : left;
31                          padding : 0;
32                          margin : 0;
33                          }
34  #lselect { border-bottom : 1px solid #fff;                   //设置#lselect 在页面中的样式
35              margin-top : 20px;
36              }
37  #larchives { border-bottom : 1px solid #fff;
38                 margin-top : 20px;
39                 }
40  #lselect h3 span, #larchives h3 span { display : none;
```

```
41                                  }
42  p { font : 12px/17px 微软雅黑;                    //设置段落文本的样式
43      margin : 0 0 17px 0;
44      }
45  a:link { font-weight : bold;                      //设置链接状态的样式
46          text-decoration : none;                   //取消链接下划线
47          color : #027d87;                          //设置链接文本的颜色
48          }
49  a:visited { font-weight : bold;                   //设置访问链接的样式
50              text-decoration : none;               //取消链接下划线
51              color : #858686;
52              }
53  a.c:visited { font-weight : normal;               //设置访问链接的样式
54                text-decoration : none;
55                color : #858686;
56                }
57  a:hover, a:active { text-decoration : underline;  //设置滑过链接的状态样式
58                      color : #000505;
59                      }
60  ul { list-style-type : none;
61      margin : 0;
62      padding : 0;
63      }
```

【深入学习】上述代码中，第 1～4 行的 linklist 样式表定义了左侧框模型的大小、位置属性。第 10～28 行代码以 preamble 开头的一系列样式表其对象是图 17.1 中的第 1 个颜色块。而第 34～41 行代码的样式表对应的是图 17.1 中的第 2 和第 3 个颜色块，从第 42～63 行是对页面基本属性的修改，如文本的颜色、链接的状态和项目列表的属性。

【运行程序】最终的显示效果如图 17.3 所示。



图 17.3　页面的左侧栏

17.2.3 页面的右侧栏主体部分

这个复杂工程的最后一项就是完成页面所剩下的右侧主栏，事实上这是最容易的一部分。在页面中，主体部分通常用来放置最重要的信息，而不需要很多花样的摆设来彰显个性。代码主体部分的写法如程序 17.7 所示。



说明：如果一个页面只有美观，而缺少内容，事实上这个页面就失去了灵魂，那么无论这个页面有多漂亮，其实不过只是一个花架子。

【代码 17-7】页面右侧主栏的结构，其源码展示如下：

程序 17.7　页面右侧主栏

```
01  <div id="supportingText">
02            <div id="explanation">
03                 <h3><span>文本内容一</span></h3>
04                 <p class="p1"><span>  域名的作用：
05  …
06                 </span></p>
07                 <p >后缀的选择：
08  …
09                 </span></p>
10            </div>
11          <div id="participation">
12                 <h3><span>文本内容二</span></h3>
13                 <p><STRONG> 什么是 robots.txt？</STRONG> </p>
14                  <p>robots.txt 是一个纯文本文件，
15  …
16                 <p >当一个搜索机器人访问一个站点时
17  …
18                  <p >robots.txt 必须放置在一个站点的根目录下，而且文件名必须全部小写。
19             </div>
20            <div id="benefits">
21                 <h3><span>文本内容三</span></h3>
22                 <p >随着网页制作经验的积累，
23  …
24            </div>
25            <div id="requirements">
26                 <h3><span>文本内容四</span></h3>
27                 <p>一般来说，绝大多数普通
28  …
29                 <p>波士顿一位图形设计者兼美术讲师说
30  …
31            </div>
```

```
32
33          <div id="footer">End
34                          </div>
35      </div>
```

【深入学习】这段代码非常容易理解，它只是将主栏部分分成一个个的段落，那么对于这样的段落，设计样式表时只要注意将文本标题和文本段落修饰好即可。它的样式表如程序 17.8 所示。

【代码 17-8】页面右侧主栏的样式表，其源码展示如下：

程序 17.8　页面右侧主栏的样式表

```
01    #supportingText {   padding : 0;                              //定义右侧第一栏的样式
02                        margin : -3px 0 40px 0;
03                        background : #d6e6b6;
04                        border : 1px solid white;
05                        width : 469px;
06                        float : right;
07                        }
08    #supportingText p { margin : 9px 17px 17px 24px;
09                        }
10    #explanation h3 { background : url(hdr_about.gif) no-repeat;    //放置右侧第一栏的标题图像
11                      width : 469px;
12                      height : 32px;
13                      margin : 0;
14                      padding : 0;
15                      }
16    #explanation h3 span { display : none;                        //隐藏对象
17                           }
18    #participation h3 { background : url(hdr_participation.gif) no-repeat; //定义右侧第二栏的样式
19                          width : 469px;
20                          height : 32px;
21                          margin : 0;
22                          padding : 0;
23                        }
24    #participation h3 span { display : none;                      //隐藏对象
25                             }
26    #benefits h3 { background : url(hdr_benefits.gif) no-repeat;   //定义右侧第三栏的样式
27                   width : 469px;
28                   height : 32px;
29                   margin : 0;
30                   }
31    #benefits h3 span { display : none;
32                        }
33    #requirements h3 { background : url(hdr_requirements.gif) no-repeat;
34                         width : 469px;
35                         height : 32px;
36                         margin : 0;
```

```
37                          padding : 0;
38                          }
39  #requirements h3 span { display : none;                    //隐藏对象
40                                }
41  #footer { background : #a2c1b9;
42           border-top : 1px solid white;
43           padding : 0 0 2px 24px;
44             }
```

【深入学习】supportingText 样式表划分了主栏在页面中的位置，以及一些基本的属性，如背景颜色和边框的样式。supportingText p 样式表则定义了主栏文本的位置属性，即在框模型中是处于一个怎样的位置。另外可以看到，#explanation h3、#participation h3、#benefits h3 和#requirements h3 这 4 个样式表是用来定义 4 段文本的标题，而这个标题使用的是图像来替代文本标题，文本则被隐藏。最后一个 footer 样式表只是简单地定位了页脚的样式，那么到这里，整个页面就完成了。



注意：第 10 行通过 url 来指定图片地址。

【运行程序】现在让我们来揭示本章开头图 17.1 的最后一层面纱，这个页面最终的显示效果如图 17.4 所示。



图 17.4　最终页面的效果

17.3　小　　结

设计页面并不是很难的一门学问，关键在于能够细心认真地处理好每一个细节，对设计保持热情，这才是保证一个设计者拥有不断创作灵感的源泉。当然，做到一个优秀的设计师没有什么捷径可走，只有通过大量的训练才能练就良好的制作感觉，才能成为一个出色的前端页面设计师。

第 18 章　综合实例三：制作英文网站

学习了关于 CSS 布局页面的方法后，如果要更好地使用各种属性，进行页面元素的控制，就要不断地实践。本章主要的内容是讲解一个完整的站点首页和一个二级页面的制作。目的是通过实例，讲解站点制作的步骤和注意的技巧。通过本章的学习，重点要掌握编写代码的方式、作站点的流程、站点结构和样式的规划等知识。

本章的主要知识点如下。

- ❑ 构思页面的布局。
- ❑ 页面的切图。
- ❑ 制作页面的顺序。
- ❑ 二级页面的制作。

【本节示例参考：资料光盘\第 18 章】

18.1　分析效果图

该实例是为了让读者更清楚使用 CSS 进行整站布局的方法，所以这里只介绍站点首页和一个二级页面的制作方法。其中，站点首页的效果图如图 18.1 所示。



图 18.1　站点首页的效果图

一个二级页面的效果图如图 18.2 所示。



图 18.2　站点二级页面的效果图

从图 18.1 和图 18.2 可以看出，首页和二级页面的头部、左侧和底部是相同的，右侧部分的宽度和内容是不同的，中间的内容部分也有很大的区别。下面分别进行分析。

1．首页效果图的分析

从图 18.1 可以看出，此时首页在纵向可以分为 3 个部分：头部（包括 logo 部分和导航）、内容部分、底部。其中，中间内容部分又可以分为 3 个部分：左侧的服务和新闻部分、中间内容部分、右侧关于我们部分。

2．二级页面的分析

二级页面和首页的结构基本相同，其中区别在于，右侧部分的宽度和首页有差别。

18.2　切　　图

分析完页面结构之后，就是切图的制作。其内容包括文本的隐藏、切片的选择、保存格式等几个方面。下面进行详细的讲解。

18.2.1　制作首页的切图

在制作切图时，首先要区分出页面的内容和修饰部分。然后分析出哪些修饰部分是可以用 CSS 代码来实现的，哪些部分是可以用重复背景来实现的，最后要切出需要知道详细宽度的部分。在制作切图时，最好把修饰背景上的文本内容去掉，同时尽量减少图片文件的数量。制作好的首页切片如图 18.3 所示。



图 18.3　首页的切片

从图 18.3 可以看出，切片中作为背景使用的大多是圆角框的部分和含有渐变颜色的部分。其中使用单纯一种颜色的部分，可以用 CSS 来实现。具体哪些修饰部分使用背景图片，哪些修饰部分使用 CSS 制作，将在后面的章节中详细介绍。



注意：切好图后，将切片保存到磁盘相应的位置。要注意的问题是，将内容部分的图片和颜色复杂的背景图片保存成 JPEG 的格式。

18.2.2　二级页面的切图

从首页和二级页面的结构可以知道，此时二级页面需要重新切图的地方并不多。其切图的主要目的是，切出两个内容图片，同时确定内容各个部分的宽度。其切图后的效果如图 18.4 所示。



图 18.4　二级页面的切图

切好图后，新建一个站点，然后把将页面中使用到的图片放入 images 文件夹中。图片的命名可以保留原有的命名，也可以重新命名，重新命名的目的就是使图片的名称更容易理解。

18.3 制作站点首页头部

切完图，新建了站点做好准备工作后，就可以开始制作页面了。同前面章节的实例制作一样，要将整个页面分成几个部分进行制作，下面分别进行讲解。

18.3.1 首页头部的信息和基础样式的制作

【代码 18-1】首先制作页面头部信息，主要包括页面标题等，其代码如程序 18.1 所示。

程序 18.1 主页头部信息

```
01  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03  <html xmlns="http://www.w3.org/1999/xhtml">
04  <head>
05  <meta name="robots" content="all" />
06  <meta name="author" content="" />
07  <meta name="Copyright" content="banquan" />
08  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
09  <meta name="description" content="" />
10  <meta content="" name="keywords" />
11  <title>Home</title>
12  <link href="style/main.css" type="text/css" rel="stylesheet" />
13  <link rel="icon" href="" type="image/x-icon" />
14  <link rel="shortcut icon" href="" type="image/x-icon" />
15  </head>
```

【深入学习】在链接样式的语句后面，第 13、14 行增加了两个 link 元素，其目的是制作收藏夹图标。



注意：第 8 行设置 charset=utf-8，如果网站全部页面都这样设置，可以防止出现乱码页面。

【代码 18-2】接下来制作页面的基础样式，其代码如程序 18.2 所示。

程序 18.2 页面的基础样式

```
01  /* 基础样式 */
02
03  *{
04          margin: 0px;
05          padding: 0px;
06          font-family: Tahoma, Verdana, Geneva, Arial, Helvetica, sans-serif;   /*定义页面使用的字体*/
```

```
07          color:#58595B;
08          font-size:11px;
09          list-style-type: none;
10          text-decoration: none;}
11    body{
12          height: 100%;
13          background-color:#5c5c5c;}               /*定义页面背景色*/
14    img{
15          border:none;}
16    a {                                             /*定义页面链接的样式*/
17          color: #ffffff;
18          text-decoration: none;}
19    a:link{
20         text-decoration:none;}
21    a:hover {
22          text-decoration: underline;}
23    form {
24          margin: 0px;
25          padding: 0px;}
26    .clear{
27         line-height:1px;
28          clear:both;
29          visibility:hidden;}
```

【深入学习】在第 3～10 行代码的基础样式中定义了字体、页面的背景颜色和相关各个页面元素的初始样式，同时取消了可能存在兼容问题的元素的补白和边界。其他都是普通样式的定义，非常简单，这里不再详述。



说明：第 15 行的 border:none，表示没有控件和边框。

18.3.2　首页头部的分析

首先还是对首页头部的效果图进行分析，其目的是区分页面中内容和修饰的部分。头部的效果图如图 18.5 所示。

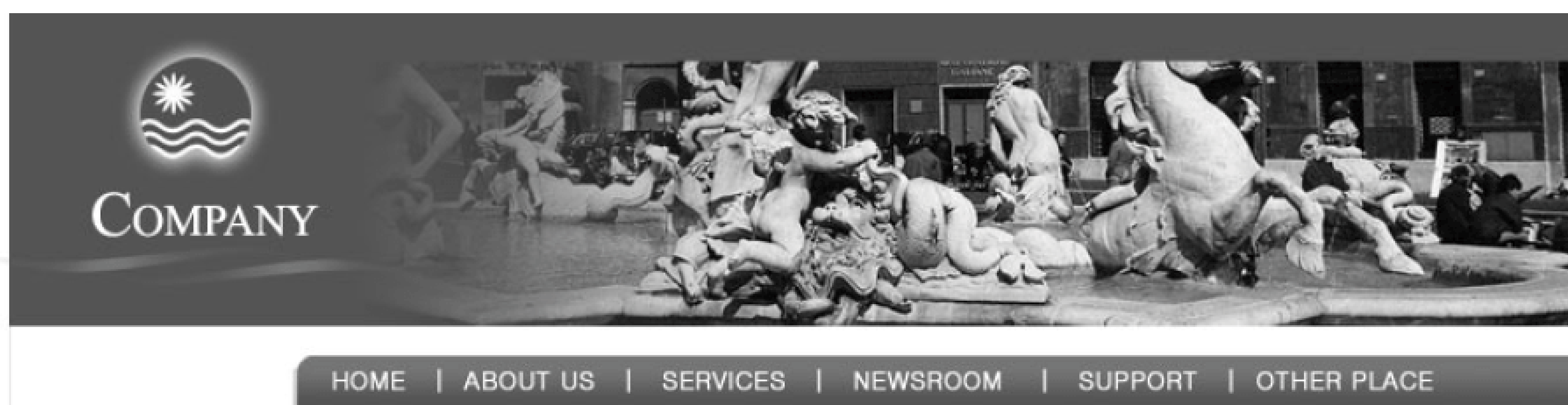


图 18.5　页面头部的效果图

从图 18.5 可以看出，头部主要分为两个部分，其中导航列表以上的部分可以用背景图片的方式实现。导航菜单部分，左侧可以用一个圆角图片背景实现，其余部分可以用一个重复的渐变背景图片实现。每个导航内容之间的白色分隔线，可以用背景图片来实现，也可以采用页面添加代码实现。

18.3.3 首页头部结构的制作

【代码 18-3】在制作头部之前，分析一下现在页面所要显示的效果。此时页面定义背景色为#5c5c5c（一种灰色），而从效果图可以看出，页面的主体部分的背景是白色。所以首先要增加一个用于显示背景颜色的父元素。下面将头部分成 header 和 menu 两个部分，分别制作，其代码如程序 18.3 所示。

程序 18.3　header和menu部分

```
01    <div id="main">                                        <!—显示白色背景的元素 -->
02    <!--head-->
03     <div  id="header">                                     <!--头部 logo 和 banner 所在的部分
-->
04          <div class="link">
05                 <a href="#">SiteMap </a>| <a href="#">Contact Us </a></div></div>
06     <div id="menu">                                        <!—导航列表开始-->
07         <div class="menulist">
08             <div class="menucontent">
09                <ul id="nav">
10                   <li><a href="#">HOME     </a></li>
11                    <li>|</li>                              <!—分隔线的部分-->
12                    <li><a href="#">ABOUT &amp; US</a></li>
13                    <li>|</li>
14                    <li><a href="#">SERVICES     </a>
15                    <li>|</li>
16                    <li><a href="#">NEWSROOM   </a></li>
17                    <li>|</li>
18                    <li><a href="#">SUPPORT    </a></li>
19                    <li>|</li>
20                    <li><a href="#">OTHER PLACE </a></li></ul>   </div>
21          </div><div class="menuleft"></div>
22          <div class="clear"></div></div></div>
```

【深入学习】第 3～5 行定义了 header 部分，这里只定义了两个链接。第 6～22 行定义了 menu 部分，其中包含一个列表，列表项都是一些导航链接。



说明：其中，menulist 元素用来显示导航列表的背景；menuleft 元素用来制作导航列表左侧圆角，分隔各个导航内容的“|”，其实是一个修饰部分。按照 CSS 布局的本质来看，应该制作成背景图片，读者可以尝试使用背景图片来实现。

18.3.4　首页头部 CSS 代码的编写

制作完页面结构之后，就可以编写 CSS 部分了。在编写 CSS 部分时，如果发现结构部分存在不合理的地方，要及时修改。

1．main 部分的样式

main 部分的样式主要作用是，制作页面白色的背景和除顶部以外的白色边界。具体代码如下所示：

```
#main{
    width:820px;
     margin:0 auto;
     background-color:#ffffff;}
```

2．header 部分的样式

header 部分的样式主要用来显示头部的背景图片，同时还要控制元素的居中显示，所以要定义元素的 margin 属性和合适的高度、宽度。同时，由于在 header 部分存在着两个导航文本，所以要控制 link 元素的位置，使导航的文本显示在正确的位置上。具体代码如下所示：

```
#header{
   width:790px;
   height:155px;
   margin:0 auto;                  /* 定义居中 */
   background:url(../images/top.jpg) no-repeat right top;}         /* 添加背景 */
.link{
   float:right;
   margin:5px 5px 0 0;     /* 精确控制链接文本的位置 */
   color:#ffffff;}
```

定义完以上样式后，页面的显示效果如图 18.6 所示。



图 18.6　定义了头部样式后的显示效果

从图 18.6 可以看出，此时头部已经显示正常了，但是下面导航列表的文本却没有了。这是由于在基础样式中定义链接的颜色为白色，同时页面的背景颜色也是白色造成的。

3．menu 部分的样式

【代码 18-4】menu 部分包括两个部分，一个是左侧的圆角框，另一个是导航列表部分。具体代码如程序 18.4 所示。

程序 18.4　menu部分的样式

```
01  #menu{
02      width:790px;
03      margin:0 auto;                  /* 居中显示 */
04      padding:10px 0 5px 0;}
05  .menulist{
06      width:620px;
07      float:right;
08      height:28px;
09      background:url(../images/index_20.gif) repeat-x;}        /* 添加列表背景 */
10  .menuleft{
11      float:right;
12      width:13px;
13      height:28px;
14      background:url(../images/index_19.gif) no-repeat;}       /* 添加圆角 */
15  .menucontent ul li{                                          /* 定义分隔线颜色和列表同行显示 */
16      color:#ffffff;
17      font-weight:bold;
18      float:left;
19      margin:5px 0 0 10px;}
20  .menucontent ul li a{
21        font-weight:bold;                                      /* 文本的加粗 */
22       color:#ffffff;
23       margin-bottom:7px;                                      /* 导航内容链接的精确定位 */
24       font-size:13px;}
25  #nav {
26        margin-left:20px;                                      /* ul 的精确定位 */
27        line-height: 16px;
28        list-style-type: none;
29        font-size:14px;}
```

【深入学习】上述代码中，首先要定义的就是 menu 元素的宽度和居中。接着要使导航列表处于 menu 元素的右侧，所以还要使用浮动属性控制导航元素的位置。



说明：为了精确定位列表的位置，还要使用相应的补白和边界属性。同时还要定义导航列表的链接样式，使导航文本能够正常显示。

【运行效果】定义了以上样式后，页面的显示效果如图 18.7 所示。

这样首页的头部就制作完成了，接下来制作首页的主体部分。



图 18.7　定义完导航部分样式后的显示效果

18.4　制作首页的主体部分

首页的主体部分可以分为 3 个部分，分别是左侧服务和新闻部分、中间内容部分、右侧的关于我们和欢迎部分。下面分别讲解它们的制作过程。

18.4.1　分析主体部分效果图

在制作之前，同样先要分析一下效果图，分清页面中的内容和修饰部分。主体部分的效果图如图 18.8 所示。

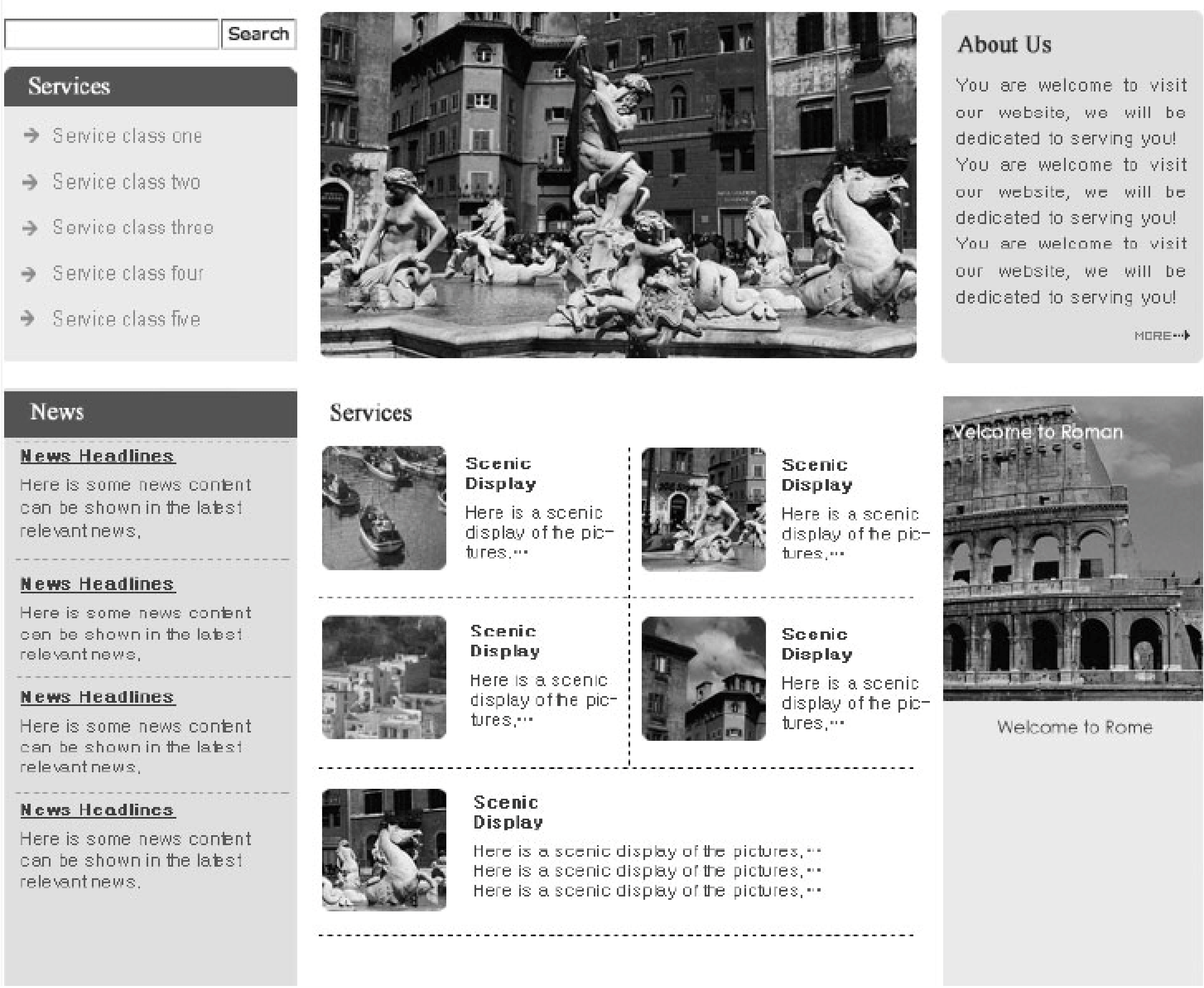


图 18.8　主体部分的效果图

从图 18.8 可以看出，左侧内容分为 3 个部分，分别为搜索部分、服务部分和新闻部分。中间内容分为两个部分，分别为展示图片部分和分类服务部分。右侧也可以分为两个部分，关于我们部分和欢迎图片部分。所以可定义 3 个浮动元素分别布局这 3 部分内容。

18.4.2 制作主体左侧部分的结构

主体左侧部分的结构可以分为下面 3 个部分来制作。

1．搜索部分的结构

【代码 18-5】搜索部分，主要由一个文本框和一个按钮构成。具体结构如程序 18.5 所示。

程序 18.5 清除浮动的元素

```
01  <div id="content">
02      <!--============左侧内容部分开始=================-->
03          <div class="left">
04              <div class="search">
05                  <form name="name1" action="" >
06                      <input type="text" size="20" name="" value="" class="botton_left"/>
07                      <input type="image" src="images/index_33.gif" name="" class="botton" /></form>
08                      <div class="clear"></div>                <!--======清除浮动元素========-->
09                      </div>
10  </div></div>
```

【深入学习】第 6 行是一个文本框，第 7 行是一个图像按钮，第 8 行用来清除浮动元素。



注意：为了控制同行显示，要使用浮动属性，为了兼容 Firefox 浏览器，所以还要添加清除浮动的元素。

2．服务列表部分的结构

【代码 18-6】服务列表部分主要由顶部的圆角、列表标题和列表内容构成。具体结构如程序 18.6 所示。

程序 18.6 服务列表

```
01  <div class="services_lefttop"></div>
02      <div class="services_lefttitle"><span class="titlewhite"><a href="#">Services</a></span></div>
03          <div class="services_leftcontent">
04              <ul>
05                  <li><a href="#">Service class one</a></li>
06                  <li><a href="#">Service class two</a></li>
07                  <li><a href="#">Service class three</a></li>
08                  <li><a href="#">Service class four</a></li>
09                  <li><a href="#">Service class five</a></li>
10                  <li><a href="#">Service class six</a></li></ul></div>
```

【深入学习】第 5～10 行使用了一个列表项，其中设置了 6 个链接。



说明：在主体结构制作中，将标题部分分成几种颜色进行独立控制，所以使用一个 span 元素来进行控制。因为页面中间部分还有 services 部分，所以在左侧部分的类名中加入了 left 字样用来区分。

3．新闻部分的结构

【代码 18-7】新闻部分也是由标题和内容两个大部分构成的，其中为了确定高度和背景，要将内容部分放到一个父元素之中。具体代码如程序 18.7 所示。

程序 18.7　新闻部分

```
01  <div class="newstitle"><span class="titlewhite">News</span></div>
02              <div class="newscontentbig">
03                 <div class="newscontent">
04                    <div class="newscontenttitle"><a href="#">News Headlines</a></div>
05  <a href="#">Here is some news content can be shown in the latest relevant news.</a></div>
06                 <div class="newscontent">                    <!--======内容部分第一条新闻========-->
07                    <div class="newscontenttitle"><a href="#">News Headlines</a></div>
08  Here is some news content can be shown in the latest relevant news.</div>
09                   <div class="newscontent">                  <!--======重复的新闻========-->
10                    <div class="newscontenttitle"><a href="#">News Headlines</a></div>
11  Here is some news content can be shown in the latest relevant news.</div>
12                    <div class="newscontent">
13                    <div class="newscontenttitle"><a href="#">News Headlines</a></div>
14  Here is some news content can be shown in the latest relevant news.</div></div>
```

【深入学习】第 4～6 行是内容部分第一条新闻。所有的新闻都放在一个名为 newscontentbig 的 div 层中。

18.4.3　制作主体左侧部分的样式

与结构部分相对应，样式部分也分为 3 个主要部分。在制作具体样式之前，首先要制作页面父元素的样式。

1．content 和 left 元素的样式

【代码 18-8】在 content 元素中，主要定义元素的宽度和居中属性，使主体内容部分和头部对齐。left 部分定义左侧内容的宽度。具体样式如程序 18.8 所示。

程序 18.8　content和left元素的样式

```
01  #content{
02      width:790px;
03      margin:0 auto 16px;
04      padding-top:5px;}
05  .left{
```

```
06        float:left;                    /*浮动属性进行定位*/
07        width:191px;}
```

2．标题部分样式

【代码 18-9】标题部分样式，分别定义在 titlewhite 和 titlered 两个类选择符中。具体样式如程序 18.9 所示。

程序 18.9　标题部分样式

```
01   .titlewhite{
02        margin-left:18px;
03        font-size:14px;
04        color:#ffffff;
05        font-weight:bold;
06        font-family: "Times New Roman", Times, serif;}          /*定义标题的字体*/
07   .titlewhite a{                                               /*定义标题含有链接时的样式 */
08        font-size:14px;
09        font-weight:bold;
10        font-family: "Times New Roman", Times, serif;}
11   .titlered{
12         margin-left:15px;
13          font-size:14px;
14          font-weight:bold;
15          color:#cc0000;
16          font-family: "Times New Roman", Times, serif;}
```

3．搜索部分样式

【代码 18-10】搜索部分的样式主要定义表单的位置、文本框的大小、按钮与文本框的间隔等属性。具体代码如程序 18.10 所示。

程序 18.10　搜索部分样式

```
01   .search{
02        margin-bottom:8px;              /*定义搜索部分与下面内容的间隔 */
03        width:191px;}
04   .search input{                       /*定义左侧的间隔 */
05        margin-left:5px;
06        height:18px;}
07   .botton_left{
08        float:left;                     /*定义浮动属性控制元素的位置 */
09        display:block;}
10   .botton{
11        float:left;
12        margin:0 0 4px 2px;}
```

4. 服务列表部分样式

【代码 18-11】服务列表部分样式主要包括定义头部圆角、标题和列表。其中包括列表的位置、间隔、字体、链接等样式。具体样式如程序 18.11 所示。

程序 18.11　服务列表部分样式

```
01  .services_lefttop{
02      width:191px;
03      background:url(../images/index_37.gif) no-repeat;        /*定义头部圆角*/
04      background-color:#ffffff;
05      height:5px;
06      font-size:0;}                                            /*取消默认高度*/
07  .services_lefttitle{
08      background-color:#006699;
09      height:20px;}                                            /*标题部分的高度*/
10  .services_leftcontent{
11      background-color:#e0edf3;                                /*定义服务列表的背景*/
12      height:140px;                                            /*定义服务列表的高度*/
13      padding:10px 0 14px 10px;}                               /*内容与顶部和底部的间隔*/
14  .services_leftcontent li{
15      margin-bottom:10px;                                      /*列表内容的间隔*/
16      font-size:12px;
17      padding-left:20px;
18      background:url(../images/ar.gif) no-repeat left;}        /*列表背景*/
19  .services_leftcontent li a{
20      color:#539CC0;
21      font-size:12px;}
```

【深入学习】在这部分的样式中，要注意的是关于宽度的部分，要保证内容的宽度不超过父元素的宽度，否则会导致页面的变形。一个可行的技巧是，尽量少定义子元素的宽度，而使用补白和边界属性进行元素的定位。因为在水平方向上，默认的宽度和边界会自动填满元素的内容。



注意：在列表中，使用背景和补白属性显示列表前修饰图片的技巧。

5. 新闻部分样式

【代码 18-12】新闻的标题部分和服务部分基本相同，采用背景颜色的方式来修饰。同时在新闻内容部分的父元素中定义合适的高度，用来显示新闻部分的背景。在新闻每一条内容中，用定义边框的样式来进行分隔。具体样式如程序 18.12 所示。

程序 18.12　新闻部分样式

```
01  .newstitle{
02      width:181px;                        /*注意宽度的计算*/
03      margin:16px 0 0;                    /*定义标题与上面内容的间隔*/
```

```
04        background-color:#006699;
05        padding:5px;}
06    .newscontentbig{
07      width:184px;
08      height:327px;
09      padding:5px 0 3px 7px;                /*定义边界与内容的间隔*/
10      background-color:#CDE3EC;}
11    .newscontent{
12      width:177px;
13      border-top:#666666 1px dashed;        /*定义虚线分隔内容*/
14      padding:3px 0 20px 0;
15      line-height:16px;}
16    .newscontent a{
17        color:#58595B;}
18    .newscontenttitle a{
19        text-decoration:underline;          /*新的链接样式*/
20        color:#024592;
21        font-weight:bold;}
```

【深入学习】此时，左侧新闻内容的具体高度并不能确定，因为要保证左侧内容、中间内容和右侧内容的高度相同，最终的高度确定，要等待其他两个部分的内容确定后才能确定。

【运行效果】定义了以上样式后，页面的显示效果如图 18.9 所示。

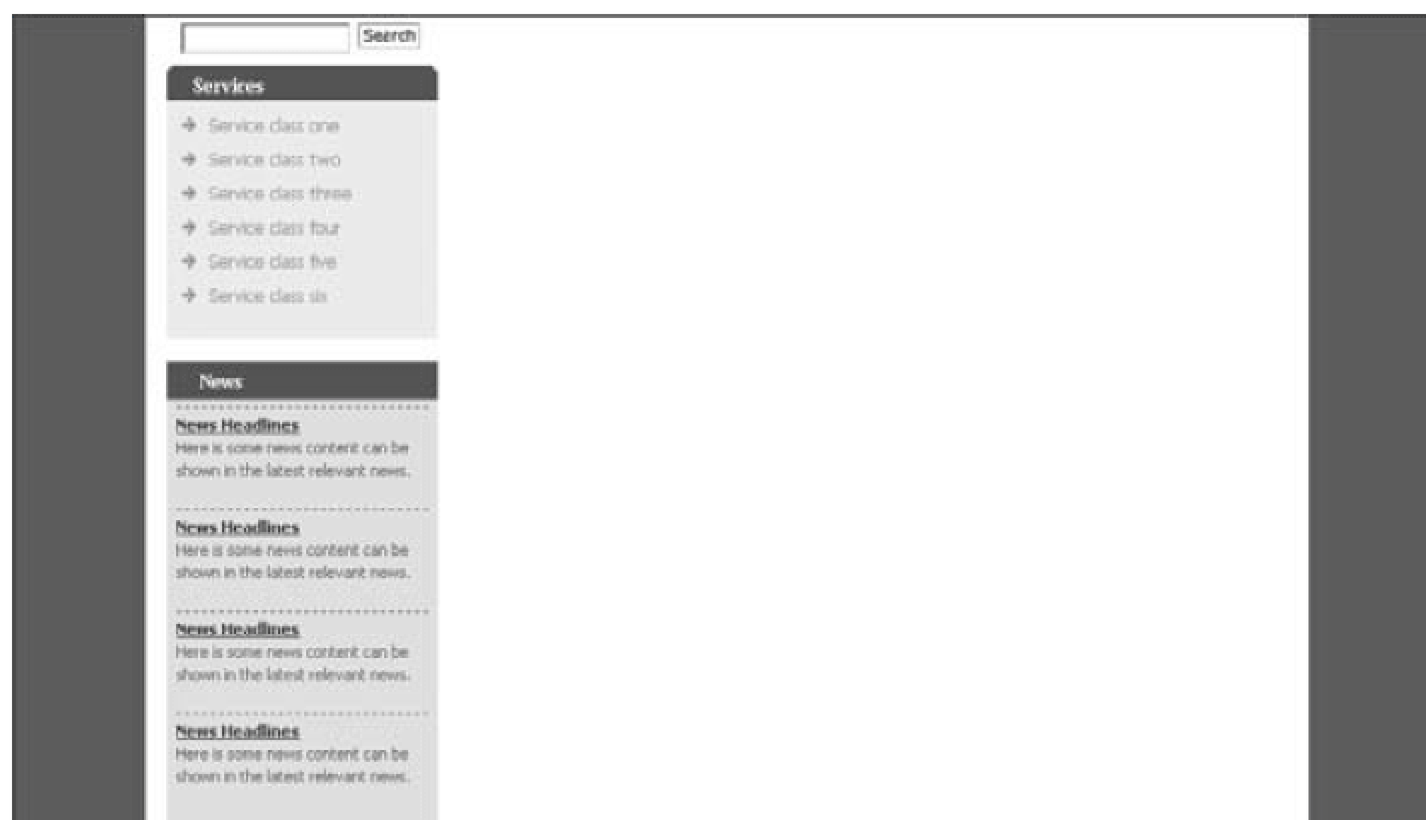


图 18.9　定义左侧内容后的显示效果

18.4.4　制作主体中间部分的结构

主体中间部分的结构可以分为两个部分：展示图片部分和服务分类部分。

1．展示图片部分的结构

【代码 18-13】展示图片部分的结构比较简单，可以不用任何包含元素，直接放在 middle 元素之中，不过由于 img 元素是内联元素，所以还要增加一个附加的 clear 元素（或者定义展示图片为块元素）来换行显示。具体的代码如程序 18.13 所示。

程序 18.13　展示图片部分的结构

```
01  <div class="middle">
02              <img src="images/show.jpg" class="middle_show" alt="pic" />
03              <div class="clear"></div>
04  </div>
```



注意：图片的宽度和高度属于图片的表现部分，所以不要定义在 img 元素中。

2．服务展示部分

【代码 18-14】服务展示部分主要由几个重复的部分组成。其中为了制作各个展示内容之间的分隔线，将 5 个展示的内容分成 3 类，左侧内容、右侧内容和底部中间内容。每个展示部分的图片、标题和内容，都使用相同的样式。具体代码如程序 18.14 所示。

程序 18.14　服务展示部分

```
01  <div class="middletitle"><span class="titlered">Services</span></div>
02          <div class="middlecontentbig">
03            <!--===========服务部分左侧内容================-->
04            <div class="middleleft">
05              <img src="images/pic1.jpg" width="81" height="81" alt="pic" />
06              <div class="piccontent">
07                <div class="pictitle"><a href="#">Scenic Display </a> </div>
08               <a href="#">Here is a scenic display of the pictures.</a> </div></div>
09                <!--===========服务部分右侧内容================-->
10            <div class="middleright">
11             <img src="images/pic2.jpg" width="81" height="81" alt="pic" />
12              <div class="piccontent">
13                <div class="pictitle"><a href="#">Scenic Display </a> </div>
14               <a href="#">Here is a scenic display of the pictures.</a></div></div>
15                 <div class="clear"></div>
16
17                 <!--===========服务部分重复内容================-->
18                 <div class="middleleft">
19              <img src="images/pic3.jpg" width="81" height="81" alt="pic" />
20              <div class="piccontent">
21                <div class="pictitle"><a href="#">Scenic Display </a></div>
22           <a href="#">Here is a scenic display of the pictures.</a></div></div>
23            <div class="middleright">
24             <img src="images/pic4.jpg" width="81" height="81" alt="pic" />
25              <div class="piccontent">
26                <div class="pictitle"><a href="#">Scenic Display </a></div>
27               <a href="#">Here is a scenic display of the pictures.</a></div></div>
28                 <div class="clear"></div>
```

```
29
30                  <!--============服务部分底部居中的内容==================-->
31                  <div class="middlecenter">
32                  <img src="images/pic5.jpg" width="81" height="81" alt="pic" />
33               <div class="piccontentcenter">
34                 <div class="pictitle"><a href="#">Scenic Display </a></div>
35                <a href="#">Here is a scenic display of the pictures.</a></div>
36                  <div class="clear"></div></div>
37           </div>
```

【深入学习】这个部分的页面内容比较多，其实并不复杂。因为每个服务项目展示的部分，都是由图片、图片标题和图片内容 3 个部分组成的。在此基础上，将同样结构的内容分别放在不同的容器中，就构成了服务展示部分的页面了。

18.4.5 制作主体中间部分的样式

对应中间部分的结构，样式部分也可以分为两个部分来讲解。在制作具体内容之前，依然先定义父元素的样式。

1. 定义 middle 元素的样式

【代码 18-15】因为首页和其他级别页面中间部分的宽度是不同的，所以此时可以不定义 middle 元素的宽度，而只定义它的浮动属性，此时 middle 元素的宽度由它所包含的元素决定。具体样式如程序 18.15 所示。

程序 18.15 新闻部分样式

```
01  .middle{
02       float:left;
03        margin-left:18px;}
04  .middle a{
05       color:#58595b;}
```



说明：该样式中，定义 margin 属性的目的是使 middle 元素中的内容部分与左侧内容之间分开一定的距离，同时定义了中间部分链接的样式。

2. 定义展示图片的样式

【代码 18-16】展示图片的样式主要是定义图片的宽度和高度。样式如程序 18.16 所示。

程序 18.16 新闻部分样式

```
01  .middle_show{
02      width:390px;
03      height:227px;}
```

3．服务展示内容部分

这个部分的样式稍微复杂一点，可以分为以下几个部分进行定义。

（1）services 标题部分。

【代码 18-17】services 标题部分是指页面中含有文本 Services 的部分。样式如程序 18.17 所示。

程序 18.17　新闻部分样式

```
01  .middletitle{
02      width:390px;
03      margin:16px 0 8px;}
```



说明：该样式主要定义了标题部分与上面图片和下面内容之间的间隔。

（2）3 个位置的容器和其中图片的样式。

【代码 18-18】处在左侧、右侧和底部中间的 3 个容器和包含图片的样式，如程序 18.18 所示。

程序 18.18　新闻部分样式

```
01  .middleleft{
02      float:left;
03      padding-bottom:5px;
04      border-right:#666666 1px dashed;                /*定义分隔线的样式*/
05      border-bottom:#666666 1px dashed;
06      width:194px;}
07  .middle img{
08      float:left;}                                    /*定义图片的位置*/
09  .middleright{
10      float:left;
11      padding-bottom:5px;
12      border-bottom:#666666 1px dashed;
13      width:194px;}
14  .middleright img{
15      margin-left:10px;}                              /*精确定义图片与分隔线间的距离*/
16  .middlecenter{
17      border-bottom:#666666 1px dashed;
18      width:390px;
19      line-height:15px;
20      padding:5px 0 10px;}                            /*控制图片的精确位置*/
```

【深入学习】这一部分的样式主要是定义各个元素的宽度，以及其中内容之间的精确分隔距离。

（3）定义图片标题和内容的样式。

【代码 18-19】其中图片标题可以使用一个统一的样式，由于底部中间部分的显示效果会略有不同，所以单独定义。具体样式如程序 18.19 所示。

程序 18.19　新闻部分样式

```
01   .piccontent{
02       float:left;
03       margin:3px auto 4px 10px;                /*精确控制内容的位置*/
04       height:80px;                             /*定义内容的高度*/
05         line-height:15px;}
06   .pictitle a{                                 /*重新定义链接的样式*/
07       color:#58595b;
08       font-size:11px;
09       font-weight:bold;}
10   .piccontentcenter{
11       float:left;
12       width:295px;                             /*定义底部中间的独立显示效果*/
13       margin-left:5px;
14       height:80px;}
```



注意：图片内容和图片的大小都是有限制的，因为此时容器的高度依赖内容的多少，过多的内容会导致页面变形。

【运行效果】定义完中间内容的样式后，页面的显示效果如图 18.10 所示。

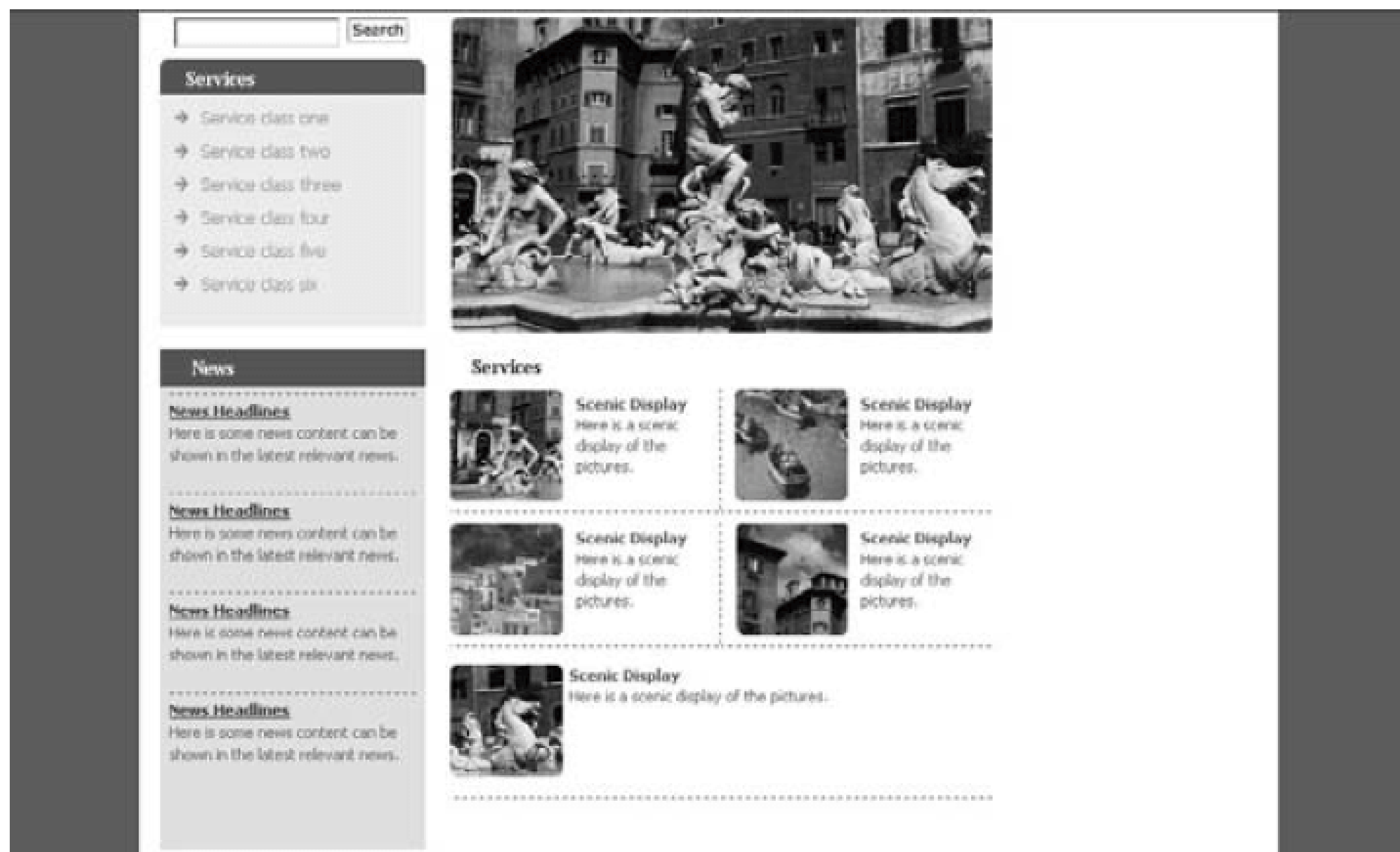


图 18.10　定义了中间内容样式后的显示效果

18.4.6　制作主体右侧部分的结构

主体右侧部分的结构可以分为两个部分，关于我们的部分和欢迎图片的部分。

1．关于我们的部分的结构

【代码 18-20】关于我们的部分的结构，大致可以分为下面几个部分：头部圆角、标题、内容、更多、底部圆角。具体的结构如程序 18.20 所示。

程序 18.20　关于我们的部分的结构

```
01  <div class="right">
02                <!--============头部圆角部分=================-->
03                 <div class="aboutustop"></div>
04                 <!--============标题部分=================-->
05                 <div class="aboutustitle"><span class="titlered">About Us</span></div>
06                 <!--============内容部分=================-->
07                 <div class="aboutuscontent">
08                You are welcome to visit our website, we will be dedicated to serving you!
09  You are welcome to visit our website, we will be dedicated to serving you!
10  You are welcome to visit our website, we will be dedicated to serving you!</div>
11                 <div class="aboutusmore">
12                    <div class="more"><a href="#">more<img src="images/index_144.gif" alt="pic" /></a></div>
13                      <div class="clear"></div></div>
14                      <!--============底部圆角=================-->
15                 <div class="aboutusbottom"></div>
16  </div>
```

【深入学习】这里通过 div 分出页面主体右侧部分的结构，上面已经介绍了包括哪些结构，这里只是提醒读者，底部圆角是通过样式来实现的。



注意：为了在 Firefox 浏览器中能够显示相同的效果，在可能用到浮动属性的地方增加了清除浮动的元素。

2．欢迎图片的部分

【代码 18-21】欢迎图片部分的结构，相对来说要简单得多，只有一个图片部分和一个欢迎文本部分。具体结构如程序 18.21 所示。

程序 18.21　欢迎图片的部分

```
01  <!--============欢迎图片=================--
02  <div class="welcomepic"><img src="images/velcome.jpg" width="171" height="202" alt="pic" /></div>
03  <!--============欢迎文本=================--
04                  <div class="welcomecontent"><a href="#">Welcome to Rome</a></div></div>
05  <!--============中间总体清除浮动元素=================--
06                 <div class="clear"></div>
```



注意：因为主体内容左侧、中间、右侧 3 个部分的定位中都使用了浮动属性，所以在最后还要添加一个清除浮动的元素，保证页面在 Firefox 浏览器中正常显示。

18.4.7 制作主体右侧部分的样式

对应右侧结构部分，依然分两个方面来定义右侧部分的样式。

1. 制作关于我们部分的样式

【代码 18-22】关于我们部分的样式和前面章节讲解的圆角框的制作方法类似。分别用背景图片制作头部和底部的圆角部分，然后用背景颜色的方法制作中间内容部分，衔接头部和底部的圆角。具体样式如程序 18.22 所示。

程序 18.22 制作关于我们部分的样式

```
01  .right{
02      float:right;}
03  .right a{
04      color:#58595B;}                          /*重新定义链接的样式*/
05  . aboutustop    {
06      width:171px;
07       height:6px;
08       background: url(../images/index_29.gif) no-repeat;     /*用背景图片制作头部圆角*/
09       font-size:0;}
10  . aboutustitle{
11      width:171px;
12       height:20px;
13       background-color:#CDE3EC;}
14  . aboutuscontent{
15      background:#cde3ec;
16       height:186px;                           /*定义容器的高度*/
17       width:151px;
18       padding:0 10px;                         /*容器内容的左右空白*/
19      line-height:18px;}
20  . aboutusmore{
21      background:#cde3ec;
22       width:171px;}
23  .more{
24      float:right;
25       margin:0 10px 10px 0;}                  /*控制文本的精确位置*/
26  .aboutusbottom{
27      width:171px;
28       height:4px;
29       font-size:0px;
30       background:url(../images/index_53.gif) no-repeat;}     /*制作底部圆角*/
```

【深入学习】这里使用的页面结构并不是最好的页面结构，这一点从制作样式表时就可以看出来。此时，样式表中定义了大量重复的宽度属性，如果给关于我们部分和欢迎部分制作一个父元素，则可以一次性地定义所有的宽度。所以，在制作过程中一定要随时分析页面结构的合理性。

【代码 18-23】定义父元素后的页面结构如程序 18.23 所示。

程序 18.23　定义父元素后的页面结构

```
01    <div class="right">
02                <!--============首页右侧的父元素==================-->
03                <div class="home_right">
04                 <!--============中间省略了原来定义的关于我们和欢迎图片部分的结构
05                ==================-->
06                   </div>
07                   <!--============首页右侧的父元素结束==================-->
08    </div>
```

【深入学习】在 home-right 选择符中定义如下样式：

```
.home_right{
   width:171px;}
```

这样，就可以去掉其子元素中所有宽度的定义了。其好处在于更改右侧内容宽度更加简单，不好的地方在于增加了页面元素。

2．制作欢迎图片部分的样式

【代码 18-24】这部分的样式比较简单，具体代码如程序 18.24 所示。

程序 18.24　制作欢迎图片部分的样式

```
01    .welcomepic{
02         margin-top:16px;}                          /*控制图片的位置*/
03    .welcomecontent{
04          background-color:#E0EDF3;
05          height:126px;                             /*控制背景的高度使其与左侧和中间基本相同*/
06          text-align:center;
07          padding: 10px;}
```

【运行效果】定义了右侧样式后，页面显示效果如图 18.11 所示。

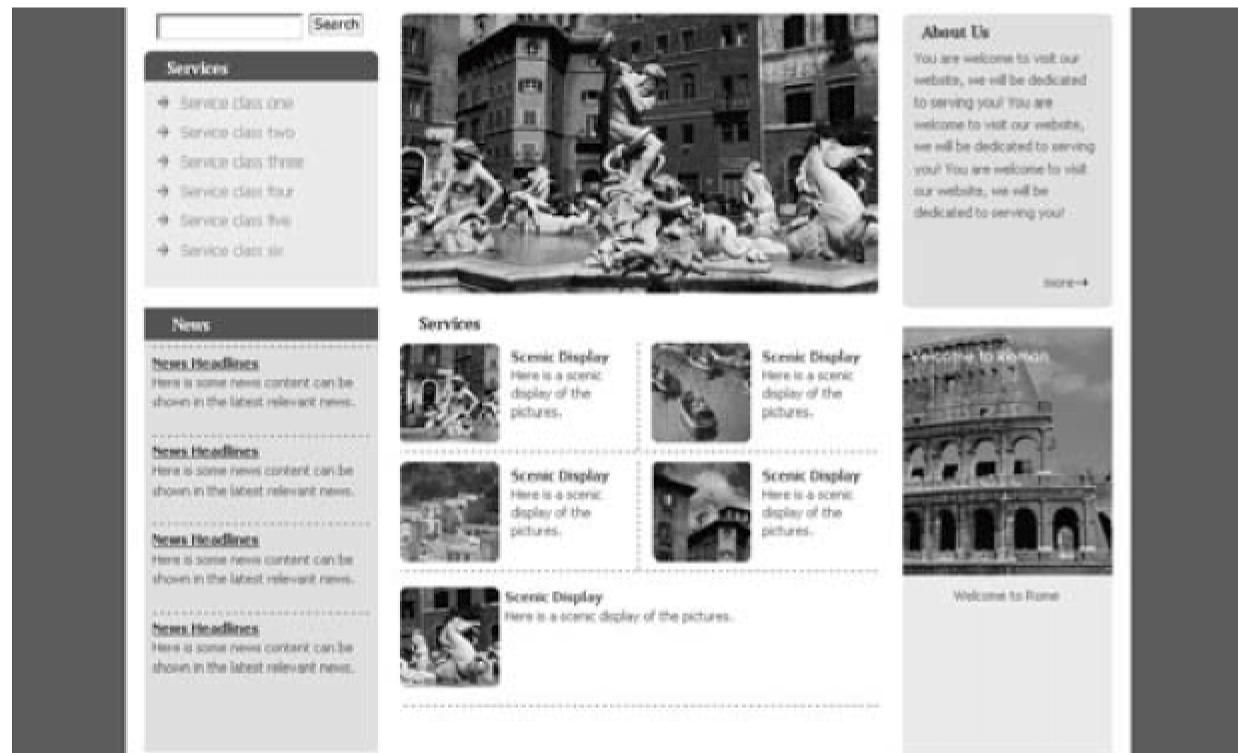


图 18.11　定义了右侧样式后的效果

18.5　制作首页的底部

首页的底部相对来说简单一些，主要由 3 个部分组成，分别是左侧的圆角、中间的内容、右侧的圆角，其效果图如图 18.12 所示。



图 18.12　底部的效果图

【代码 18-25】底部的页面结构如程序 18.25 所示。

程序 18.25　底部的页面结构

```
01  <div class="footer">
02          <div class="footerleft"></div>
03           <div class="footercontent">
04               <div class="footercontentleft">Welcome to ******.com</div>
05                <div class="footercontentright">2006 International</div></div>
06           <div class="footerright"></div></div>
```

【深入学习】结构很简单，左右两侧是用来制作圆角的，中间的内容又分成左右两个部分。

【代码 18-26】上述代码的样式如程序 18.26 所示。

程序 18.26　底部的页面结构的样式

```
01  .footer{
02    margin:0 auto;                                    /*定义父元素的居中*/
03    width:790px;
04    height:36px;
05    padding-bottom:5px;}
06  .footerleft{
07     float:left;
08     background:url(../images/index_83.gif) no-repeat left;     /*制作左侧圆角*/
09     width:5px;
10     height:26px;}
11  .footercontent{
12     float:left;
13     width:780px;
14     height:26px;
15     background-color:#006699;}                   /*内容部分的背景*/
16  .footercontentleft{
17     float:left;
18     margin:3px 0 0 10px;
19     color:#ffffff;
20     font-weight:bold;}
```

```
21   .footercontentright{
22       float:right;
23       margin:3px 0 0 10px;
24       color:#ffffff;
25       font-weight:bold;}
26
27   .footerright{
28       float:left;
29       background:url(../images/index_86.gif) no-repeat right;        /*制作右侧圆角*/
30       width:5px;
31       height:26px;}
```

【深入学习】第 2 行是定义父元素的居中，这种方法在界面设计中常常碰到。第 8 行和第 29 行制作的是圆角界面。

【运行效果】此时页面底部的显示效果如图 18.13 所示。

Welcome to ******.com　　2006 International

图 18.13　定义样式后的底部效果

18.6　首页的兼容问题

以上的制作过程都是在 IE 6.0 下进行的，制作好的首页在 Firefox2.0 中的显示效果如图 18.14 所示。



图 18.14　首页在 Firefox2.0 中的显示效果

从图 18.14 中可以看到，此时存在的显示问题出现在中间服务展示的部分。从页面效果来看，很可能是内容宽度的问题，也可能是由于并列的浮动元素在 Firefox 浏览器中的特殊性造成的。

【代码 18-27】首先看一下图片和内容部分的结构（以左侧部分为例），如程序 18.27 所示。

程序 18.27　图片和内容部分的结构

```
01   <div class="middleleft">
02                  <img src="images/pic1.jpg" width="81" height="81" alt="pic" />
03                  <div class="piccontent">
04                    <div class="pictitle"><a href="#">Scenic Display </a> </div>
05                         <a href="#">Here is a scenic display of the pictures.</a> </div>
06   </div>
```

【深入学习】上述代码从结构上看，内容和图片是属于同级的关系。

【代码 18-28】图片和内容部分的结构的样式如程序 18.28 所示。

程序 18.28　图片和内容部分的结构的样式

```
01   .middle img{
02          float:left;}
03   .piccontent{
04        float:left;
05        line-height:15px;
06        margin:3px 0 4px 10px;
07        height:80px;
08        background:#999999;}
```

【深入学习】从定义的样式中可以看到，此时内容部分没有定义宽度，但是两个元素都定义了浮动属性。所以可以分析出，此时的问题是由于并列浮动元素在 Firefox 中没有定义宽度造成的。所以在 piccontent 中增加宽度代码如下所示：

```
width:90px;
```

18.7　二级页面的制作

从效果图中可以看出，首页和二级页面的头部、左侧、底部都是相同的，所以只需要更改首页中间内容部分和右侧部分即可。

18.7.1　分析二级页面的效果图

二级页面的中间内容部分的效果图如图 18.15 所示。

从图 18.15 可以看出，此时中间内容部分是由上面的展示图片和下面的新闻列表组成的。新闻列表部分又分为日期和内容两个部分，右侧部分是由一个列表和一个图片组成。下面详细讲解具体的制作过程。

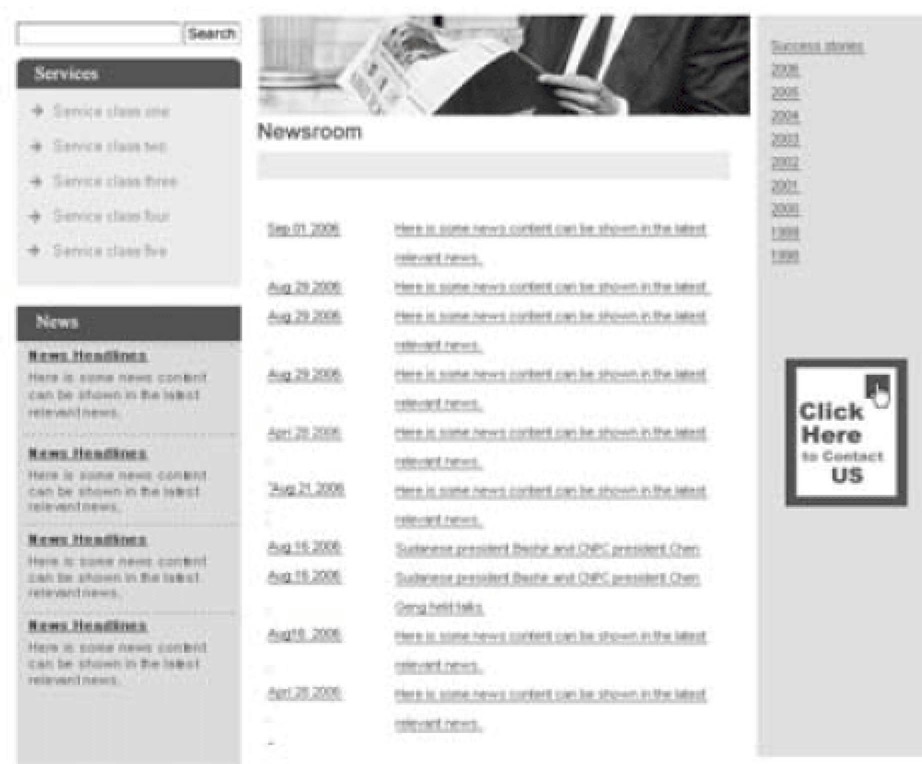


图 18.15　二级页面内容部分的效果图

18.7.2　制作二级页面中间内容部分的结构

在制作二级页面之前，首先要在页面头部文件中增加链接新样式表的代码，如下所示：

```
<link href="style/newsroom.css" type="text/css" rel="stylesheet" />
```

【代码 18-29】中间图片展示的部分可以使用首页的结构，不同之处在于，此时图片使用的样式不同，因为此时图片的宽度和首页图片的宽度不同。接下来的部分是一个标题，然后是分隔的颜色块，再下面是日期和标题的列表。具体的代码如程序 18.29 所示。

程序 18.29　中间内容部分的结构

```
01  <div class="middle">
02  <img src="images/newsroom.jpg" alt="event" />
03              <div class="clear"></div>
04  <!--============标题部分=================-->
05          <div class="two_middletitle"><span class="titleblue">Newsroom</span></div>
06  <!--============分隔色块=================-->
07          <div class="spaceline"></div>
08  <!--============新闻内容部分=================-->
09          <div class="two_middlecontentbig">
10            <div class="left_list">
11                <ul>
12                    <li>Sep 01 2006</li>
13                    <li>Aug 29 2006</li>
14                    <li>Aug 29 2006</li>
15                    <li>Aug 29 2006</li>
16                    <li>Aug 29 2006</li>
17                    <li>Aug 29 2006</li>
18                    <li>Aug 29 2006</li>
19                    <li>Aug 29 2006</li>
20              </ul></div>
21             <div class="right_list">
```

```
22                      <ul>
23                          <li><a href="#">
24                            Here is some news content can be shown in the latest relevant news.</a></li>
25                          <li>Here is some news content can be shown in the latest relevant news</li>
26                          <li>Here is some news content can be shown in the latest relevant news</li>
27                          <li>Here is some news content can be shown in the latest relevant news</li>
28                          <li>Here is some news content can be shown in the latest relevant news.</li>
29                          <li>Here is some news content can be shown in the latest relevant news</li>
30                          <li>Here is some news content can be shown in the latest relevant news</li>
31                          <li>Here is some news content can be shown in the latest relevant news</li></ul></div>
32                        <div class="clear"></div>
33                        </div>
34     </div>
```

【深入学习】第 10～20 行是左侧列表，按时间排列；第 21～31 行是右侧列表，显示新闻内容。



注意：因为新闻内容部分是用程序显示的，所以，不用将所有的含有链接的列表都制作成空的链接（空链接的意思是路径为“#”的链接）。

18.7.3 制作二级页面中间内容部分的样式

这个部分的样式可以分为两个部分，一部分是其他二级页面也会使用的公用样式，另一部分是新闻页面独立使用的样式。其中，公用样式要定义在 mian.css 中，独立的样式定义在 newsroom.css 中。下面分别进行制作。

1. 制作页面中间内容公用的样式

【代码 18-30】二级页面的公用样式包括图片大小、标题、分隔色块等，其样式如程序 18.30 所示。

程序 18.30 页面中间内容公用的样式

```
01   .titleblue{                                          /*所有二级页面内容部分使用的标题*/
02         margin-left:10px;
03          font-size:18px;
04          color:#006599;
05          font-family:Arial, Helvetica, sans-serif;}
06   .two_showpic{                                        /*二级页面展示图片的大小*/
07         width:416px;
08          height:87px;}
09   .two_middletitle{                                    /*二级页面标题与图片的距离*/
10      margin-top:16px;}
11   .spaceline{                                          /*二级页面中的分隔色块*/
12       height:22px;
13       width:400px;
14       margin:6px 0 10px 10px;
15       background-color:#E0EDF3;}
```

【深入学习】因为所有的二级页面都将使用以上样式，所以可定义在 main.css 中，便于其他二级页面的调用。

2．制作页面中间内容独立的样式

【代码 18-31】新闻页面中间部分的独立样式，主要是对两个列表的控制。其中包括列表的宽度、文本、链接等属性，其具体样式如程序 18.31 所示。

程序 18.31　页面中间内容独立的样式

```
01  .left_list{                                   /*使用浮动属性对左侧列表定位，同时定义宽度*/
02         float:left;
03          width:120px;
04         margin-left:6px;}
05  .left_list li{                                /*定义列表的高度等属性*/
06       color:#006599;
07       height:52px;
08       line-height:20px;}
09  .right_list{                                  /*使用浮动属性控制右侧列表与左侧列表同行显示*/
10       width:280px;
11       float:left;}
12  .right_list li{
13       height:52px;
14       line-height:20px;}
15  .right_list li a{                             /*定义链接的颜色*/
16       color:#58595b;}
```

【深入学习】列表部分的样式，主要是通过使用补白、边界和行高等属性，将各个列表元素区分开。

【运行效果】页面在定义完中间部分样式后，显示效果如图 18.16 所示。

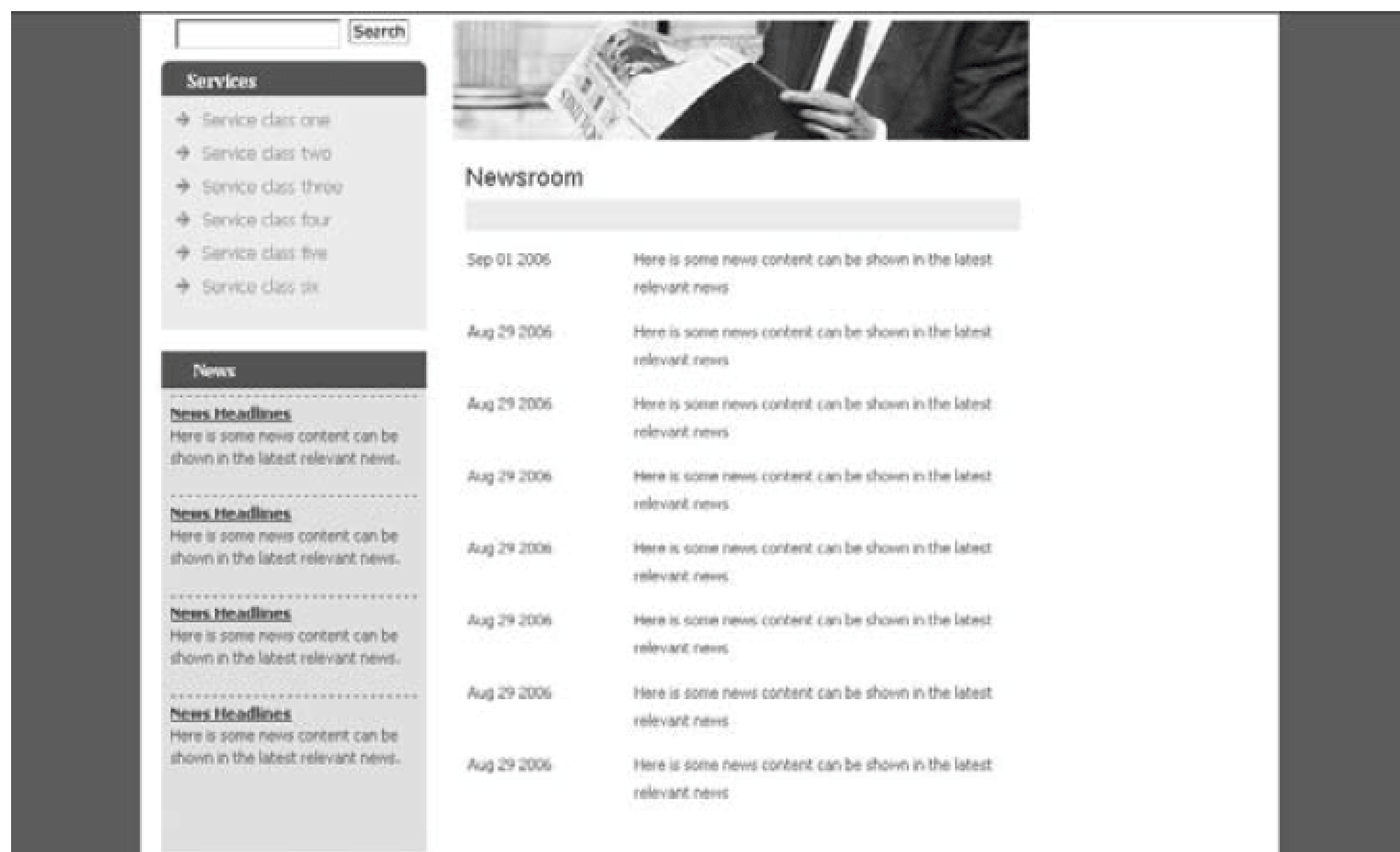


图 18.16　定义完页面新闻内容部分样式后的显示效果

18.7.4 制作二级页面右侧部分的结构

【代码 18-32】二级页面右侧部分的结构比较简单，包括一个标题、一个列表和一个图片。具体的代码如程序 18.32 所示。

程序 18.32 二级页面右侧部分的结构

```
01  <div class="right">
02      <div class="two_rightcontent">
03          <div class="press_title"><a href="#">Success stories</a></div>   <!--===标题部分===-->
04          <ul>                                                             <!--===列表部分===-->
05              <li><a href="#">2006</a></li>
06              <li><a href="#">2005</a></li>
07              <li><a href="#">2004</a></li>
08              <li><a href="#">2003</a></li>
09              <li><a href="#">2002</a></li>
10              <li><a href="#">2001</a></li>
11              <li><a href="#">2000</a></li>
12              <li><a href="#">1999</a></li>
13              <li><a href="#">1998</a></li></ul>
14                                                                           <!--===图片部分===-->
15          <a href="#"><img src="images/contact.jpg" width="104" height="129" alt="pic" /></a>
16      </div>
17  </div>
```

【深入学习】第 5～13 行是一个列表，第 15 行是个图片，这个图片的文件地址通过 src 属性决定。



说明：在定义结构时，要合理利用元素的包含关系，尽量减少页面元素的数量。

18.7.5 制作二级页面右侧部分的样式

【代码 18-33】因为整个二级页面的右侧内容都将使用现在的内容，所以要把右侧的样式都定义在 main.css 文件中。具体样式如程序 18.33 所示。

程序 18.33 二级页面右侧部分的样式

```
01  .two_rightcontent{                          /*定义右侧内容的高度和背景*/
02      background-color:#cde3ec;
03      width:136px;
04      height:590px;
05      padding:10px 10px 0;}                   /*定义内容与边界的间隔*/
06  .two_rightcontent img{
07      margin:40px 0 0 16px;}                  /*定义图片的位置*/
08  .two_rightcontent li{
```

```
09        line-height:20px;}                               /*定义列表的间隔*/
10    .two_rightcontent a{
11        color:#006699;                                   /*定义新的链接样式*/
12         text-decoration:underline;
13         font-size:11px;}
```

【深入学习】上述代码通过注释可以看出，第 8、9 行定义的是列表的间隔，这样是为了让列表看起来更舒服、更美观。

【运行效果】页面在定义右侧内容后，显示效果如图 18.17 所示。

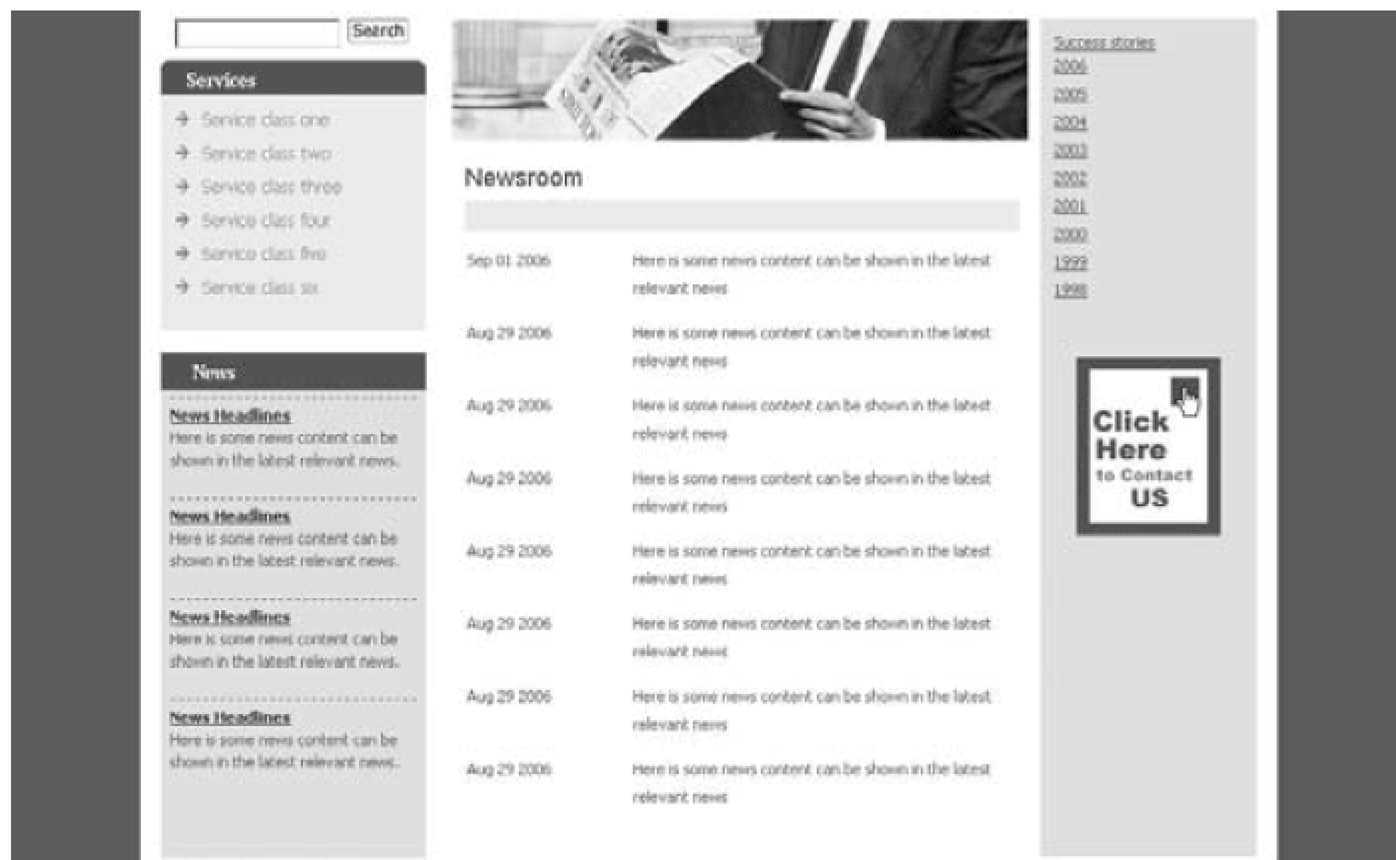


图 18.17　定义完右侧样式后的显示效果

在 Firefox 中进行测试，页面显示效果相同。这样，二级页面就制作完成了。

18.8　小　　结

本章虽然是一个英文网站，但整个分析思路和中文网站并没有什么区别。本章首先介绍如何进行主页面的切图，然后介绍如何制作首页的头部，其次是首页的主体部分，最后才是首页的底部。制作完主页后，本书才开始介绍二级页面的制作。希望读者通过本章的分析思路能够掌握网站的制作流程和技巧。

第 19 章　综合实例四：使用 Dreamweaver 制作中文网站

在制作网页时，使用可视化的开发工具，不但简单易用，而且可以大大提高站点开发的效率。本章的主要内容是讲解如何使用可视化开发工具 Dreamweaver 进行标准网站的开发，其中包括一个完整的站点首页和一个二级页面的制作。通过本章的学习，重点要掌握使用 Dreamweaver 添加页面元素和样式的方法与用可视化开发工具制作页面时需要注意的问题等知识。

本章的主要知识点如下。

- ❑ 效果图的分析。
- ❑ 站点首页的制作。
- ❑ 二级页面的制作。
- ❑ 兼容问题及解决。

【本节示例参考：资料光盘\第 19 章】

19.1　分析效果图

同样，在本实例中也只讲解站点首页和一个二级页面的制作方法，其他页面的制作可以参照类似的方法。其中，站点首页的效果图如图 19.1 所示。



图 19.1　站点首页的效果图

一个二级页面的效果图如图 19.2 所示。



图 19.2　站点二级页面的效果图

从图 19.1 和图 19.2 可以看出，首页和二级页面的头部、左侧和底部是相同的，右侧部分的宽度和样式也是相同的，区别在于内容不同。

1．首页效果图的分析

从图 19.1 可以看出，首页在纵向可以分为 3 个部分：头部（包括 logo 部分和导航）、内容部分、底部。其中，中间内容部分又可以分为两个部分：左侧的公告热点信息等，右侧的关于我们、新闻列表和相关链接的部分。

2．二级页面的分析

二级页面和首页的结构基本相同，区别在于右侧的内容此时为新闻列表。

19.2　制作首页的切图

分析完页面结构后，就要进行切图了。同样要注意文本的隐藏、切片的选择、保存格式等几个方面。下面进行详细的讲解。

在制作切图时，首先要区分出页面内容和修饰的部分，然后分析出哪些修饰部分是可以用 CSS 代码来实现的，哪些部分是可以用背景图片来实现的，哪些是需要知道详细宽度的。在制作切图时，首先要把影响背景的文本内容去掉，同时要尽量减少图片文件的数量。制作好的首页切片如图 19.3 所示。



图 19.3　首页的切片

图 19.3 的切片中，用作背景的图片包括头部背景、导航背景、主体背景、底部背景、分类图标，其他的图片为内容图片。切好图后将切片保存到磁盘相应的位置，因为实例中二级页面内容部分没有新的图片，所以可以不进行切图操作。新建一个站点，然后将使用到的图片放入 images 文件夹中。

19.3　制作站点首页头部

做好准备工作后，就可以开始制作页面了。同前面章节的实例制作一样，首页头部也要分成几个部分进行制作。下面分别进行讲解。

19.3.1　首页头部的信息和基础样式的制作

首先建立 index.html 页面。关于建立文件的方法，前面章节已经讲解过了，这里不再重述。然后制作链接的样式文件。

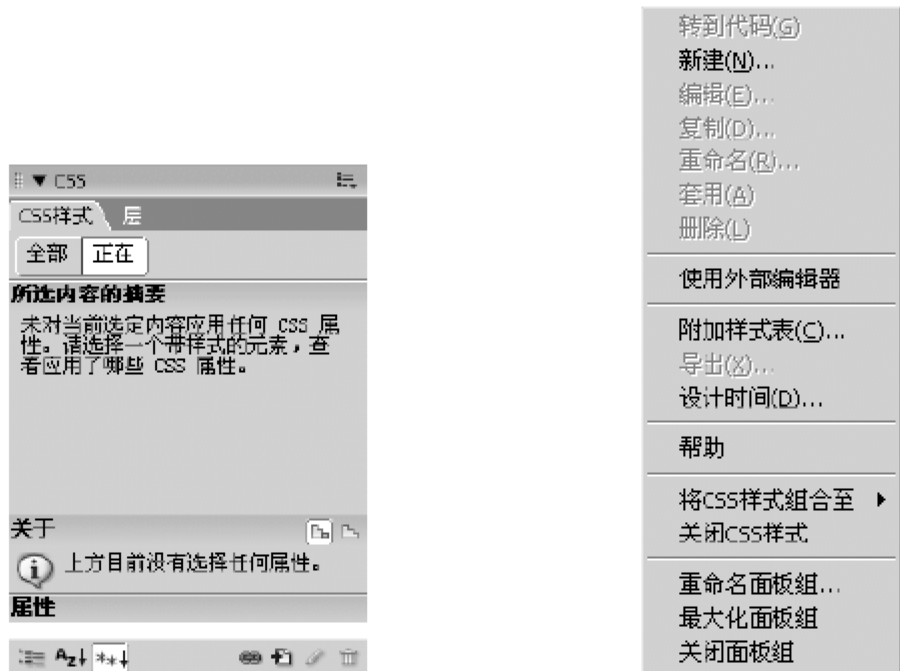
1．制作链接的外部样式文件

（1）选择“文件”|“新建”命令，新建一个 css 文件。

（2）选择“文件”|“另存为”命令，将新建的 css 文件保存在 style 文件夹中，并命名为 main.css。

（3）选择“窗口”|“CSS 样式”命令，打开 CSS 控制面板，如图 19.4 所示。单击 CSS 控制面板顶端右侧的按钮，打开下拉子菜单，如图 19.5 所示。

（4）选择“附加样式表”命令，打开“链接外部样式表”对话框，选择制作的 main.css 文件，如图 19.6 所示。





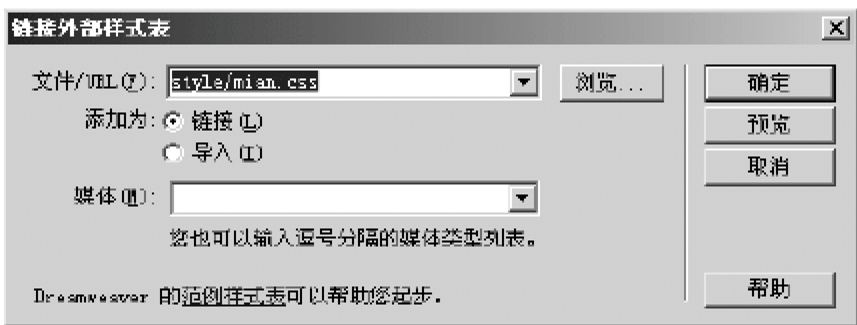


图 19.4　CSS 控制面板　　图 19.5　CSS 控制面板子菜单　　图 19.6　“链接外部样式表”对话框

单击“确定”按钮，制作好链接的外部样式。在代码视图中，对应的代码如下所示：

```
<link href="style/mian.css" rel="stylesheet" type="text/css" />
```

2．设置页面属性

（1）选择“修改”|“页面属性”命令，打开“页面属性”对话框，如图 19.7 所示。页面属性有 5 个分类，其中常用的是定义页面的外观、链接、标题、标题/编码。在本实例中，所选用的外观属性参数如图 19.8 所示。





图 19.7　“页面属性”对话框　　图 19.8　关于外观的设置

（2）在本实例中，所选用的链接属性参数如图 19.9 所示。标题是指页面中 h1 到 h6 的标题元素的样式，如果页面中不需要，可以不设置相关属性。

（3）标题/编码是指页面的标题，默认的设置是“无标题文档”，所以要重新定义页面标题。关于编码，中文常用的是 gb2312 编码，也是页面默认的编码。具体的参数设置如图 19.10 所示。





图 19.9　链接的参数设置　　图 19.10　标题/编码的参数设置

（4）单击“确定”按钮，这样页面的属性就定义完了。

【代码 19-1】不过此时定义页面属性所产生的 CSS 代码，会显示在页面 head 元素中，其在代码视图中显示的代码如程序 19.1 所示。

程序 19.1　定义页面属性所产生的CSS代码

```
01  <style type="text/css">
02  <!--
03  body,td,th {
04          font-family: 宋体;
05          font-size: 12px;
06          color: #dddddd;
07  }
08  body {
09          background-color: #044cba;
10          margin-left: 0px;
11          margin-top: 0px;
12          margin-right: 0px;
13          margin-bottom: 0px;
14  }
15  a {
16          font-size: 12px;
17  }
18  a:link {
19          text-decoration: none;
20  }
21  a:visited {
22          text-decoration: none;
23  }
24  a:hover {
25          text-decoration: none;
26          color: #FFFF66;
27  }
28  a:active {
29          text-decoration: none;
30          color: #FFFF66;
31  }
32  -->
33  </style>
```

【深入学习】显然，这种在页面中调用 CSS 的方式并不方便。因为使用这种方式，在新制作的每一个页面中都要重新定义页面属性。所以要将 style 元素中的 CSS 代码粘贴到 main.css 文件中，以方便其他页面的调用。同时取消 style 元素的定义。

【代码 19-2】经过更改后，页面头部的代码如程序 19.2 所示。

程序 19.2　页面头部的代码

```
01  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03  <html xmlns="http://www.w3.org/1999/xhtml">
04  <head>
05  <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
06  <title>css 实例 2</title>
07  <link href="style/mian.css" rel="stylesheet" type="text/css" />
08  </head>
```

19.3.2　首页头部的分析

首先还是对首页头部效果图进行分析，主要目的是区分页面中内容和修饰的部分。头部的效果图如图 19.11 所示。



图 19.11　页面头部的效果图

从图 19.11 可以看出，头部主要分为两个部分，其中导航列表以上部分可以用背景图片的方式实现，但是由于图片比较大，所以分两个部分来显示（目的是提高图片的加载速度）。下面是一个导航菜单，因为使用了一个大的背景，所以只要控制好导航列表的显示位置就可以了。

19.3.3　首页头部 logo 和 banner 部分的制作

首页头部结构比较简单，主要由两个用来显示背景的元素和一个用来显示列表的元素组成的。其中，导航列表以上的内容分成两个部分，分别是 logo 部分和 banner 部分。下面分别讲解详细的制作过程。

（1）在 Dreamweaver 界面的设计视图中选择“插入”|“布局对象”|“Div 标签”命令，打开“插入 Div 标签”对话框，并添加相应的参数，如图 19.12 所示。



说明：在插入选项中有 3 种选择。“在插入点”选项的意思是，在光标所在的位置插入新的元素。

（2）单击“新建 CSS 样式”按钮，打开“新建 CSS 规则”对话框，并设置相应的参数，如图 19.13 所示。



图 19.12　添加 header 元素

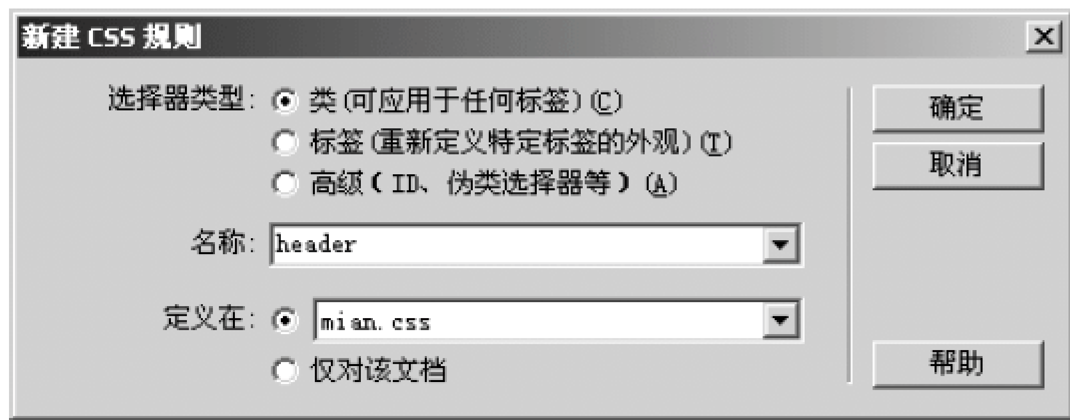


图 19.13　新建 header 元素的 CSS 规则

（3）单击“确定”按钮，打开“CSS 规则定义”对话框，如图 19.14 所示。在“CSS 规则定义”对话框的“分类”栏中，将 CSS 属性分为 8 类，现分别介绍如下。

- 类型：主要用来定义文本的属性。包括字体、字体的大小、字体的样式、加粗、行高、修饰、颜色等属性。
- 背景：主要用来定义元素的背景属性。包括背景颜色、背景图片、背景附件、背景的重复和位置等属性。
- 区块：主要用来定义文本的缩进和对齐属性。包括水平对齐、垂直对齐、文本的缩进、空白的设置等属性。
- 方框：主要用来定义元素的除边框外的盒模型区域和浮动属性。包括宽度、高度、补白、边界、浮动、清除等属性。
- 边框：主要用来定义边框的属性。包括边框宽度、边框样式、边框颜色等属性。
- 列表：主要用来定义列表的相关属性。包括列表类型、项目符号替换图片、位置等属性。
- 定位：主要用来定义定位属性。包括绝对定位、相对定位、固定定位，以及各个方向上的边偏移属性等。
- 扩展：主要用来定义一些光标显示、分页、滤镜等属性。



说明：在“CSS 规则定义”对话框中，除扩展外使用的大多数属性，本书都已经做过详细介绍。所以在使用 Dreamweaver 定义元素属性时，只需要在各个分类的对话框中添加相应的参数即可。

（4）ID 选择符 header 中，设置的 CSS 属性如图 19.14、图 19.15 所示。



图 19.14　“CSS 规则定义”对话框



图 19.15　header 元素的方框属性

（5）此时，header 元素只定义了方框属性，其中所添加的参数分别是宽度为 998px。边界属性中，

上下边界为 0，左右边界为 auto，目的是使元素水平居中显示。单击“确定”按钮，完成 header 元素的样式。在页面中删除软件默认添加的提示内容，然后仿照添加 header 元素的方法添加其他的元素。

1．制作 logo 元素的样式

（1）选择“插入”|“布局对象”|“Div 标签”命令，打开“插入 Div 标签”对话框，并添加相应的参数，如图 19.16 所示。然后单击“新建 CSS 样式”按钮定义 logo 的样式，其具体的参数如图 19.17、图 19.18 所示。



图 19.16　添加 logo 元素



图 19.17　logo 元素的背景属性

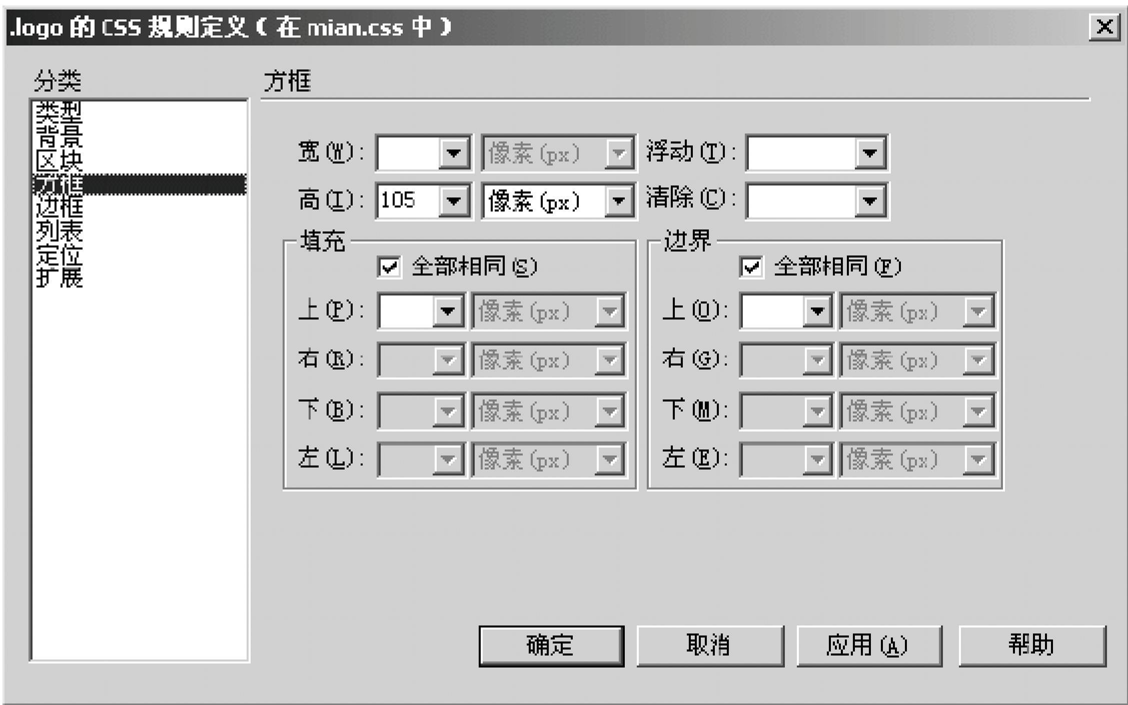


图 19.18　logo 元素的方框属性

（2）定义完以上样式，同时去掉软件自动生成的内容后，页面的显示效果如图 19.19 所示。



图 19.19　定义完 logo 元素样式后的显示效果

2．定义 banner 元素的样式

（1）选择“插入”|“布局对象”|“Div 标签”命令，打开“插入 Div 标签”对话框，并添加相应的参数，如图 19.20 所示。



图 19.20　添加 banner 元素



注意：在添加新元素之前，一定要把光标放置在相应的位置，例如，现在添加的 banner 元素在 logo 元素的后面，所以可以使用键盘上向右的方向键，将光标移动到 logo 元素之外，再使用如图 19.20 所示的参数，添加 banner 元素。如果在设计视图中无法看出光标的具体位置，可以到代码视图中确认。

（2）单击“新建 CSS 样式”按钮定义 banner 的样式，其具体的参数如图 19.21、图 19.22 所示。



图 19.21　banner 元素的背景属性



图 19.22　banner 元素的方框属性

（3）定义了 banner 元素的样式，同时去掉软件自动生成的内容后，页面的显示效果如图 19.23 所示。



图 19.23　定义完 banner 元素样式后的显示效果

19.3.4　导航列表的制作

导航列表是由两个部分组成的，分别用来显示背景的父元素和导航内容的列表元素，其具体的制作方法如下所示。

1. 父元素 menu 的制作

将光标移动到 banner 元素之外，选择“插入”|“布局对象”|“Div 标签”命令，打开“插入 Div 标签”对话框，在相应的参数中添加类名称为 menu。单击“新建 CSS 样式”按钮定义 menu 的样式，其具体的参数如图 19.24、图 19.25 所示。



图 19.24　menu 元素的背景属性



图 19.25　menu 元素的方框属性

2. 列表元素的制作

（1）将光标移动到 menu 元素之内，选择“插入”|HTML|“文本对象”|“项目列表”命令，添加项目列表。然后选择“插入”|HTML|“文本对象”|“列表项”命令，添加列表内的项目，同时添加内容“企业形象页”。调整光标，依次添加其余的列表项和内容，此时，Dreamweaver 会自动切换到拆分视图。在拆分视图中，选择 ul 元素及其包含的 li 元素并右击，在弹出的快捷菜单中选择“CSS 样式”|“新建”命令，打开“新建 CSS 规则”对话框，此时对话框中将会有默认的选择符，如图 19.26 所示。

（2）使用默认的参数，单击“确定”按钮，打开“CSS 规则定义”对话框，其中定义的样式如图 19.27、图 19.28 所示。

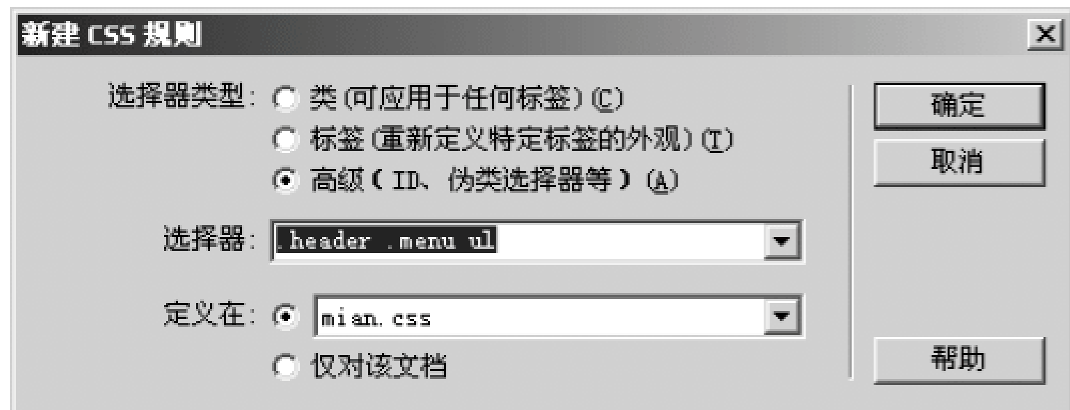


图 19.26　新建 CSS 规则的默认参数



图 19.27　定义列表的方框属性



图 19.28　定义列表的类型

（3）在拆分视图中选中所有的 li 及其内容，使用和新建列表样式一样的方法建立列表项的样式，其具体参数如图 19.29、图 19.30 所示。

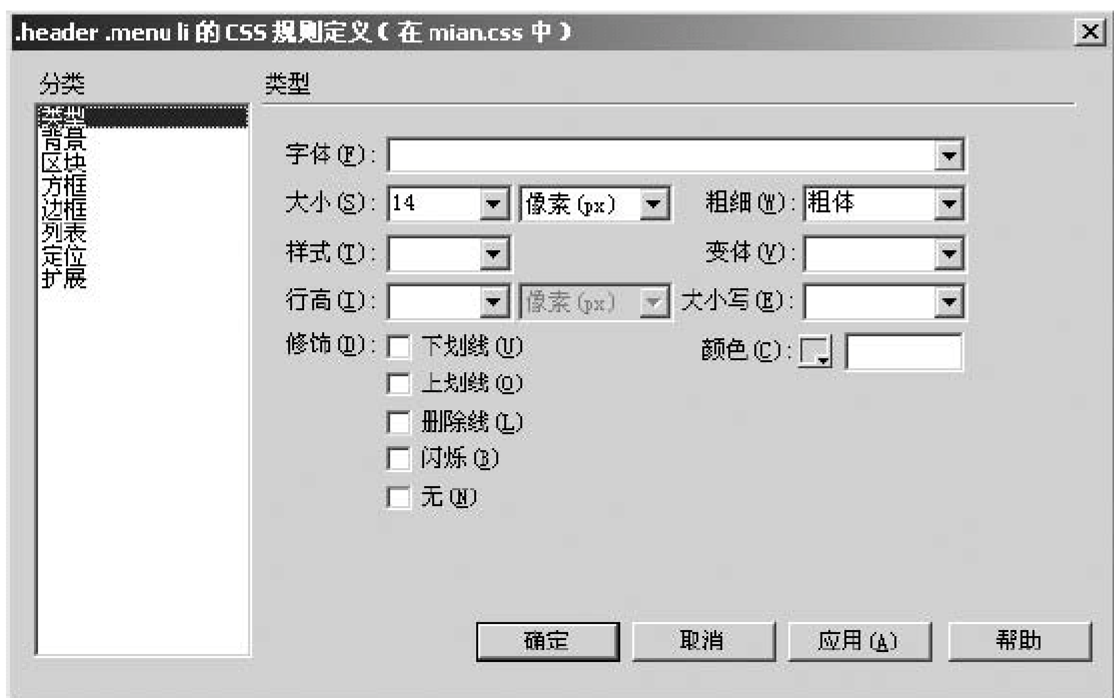


图 19.29　定义列表的类型属性



图 19.30　定义列表的方框属性

（4）以上定义的列表属性的参数是随意添加的，定义后的显示效果如图 19.31 所示。



图 19.31　定义了列表属性之后的显示效果

（5）从图 19.31 可以看出，此时列表存在的主要问题有两个方面，一个是列表的位置不对，另一个是列表内容之间的间隔过宽。所以要重新更改列表和列表内容属性，更改的方法是，单击 CSS 控制面板上的“全部”按钮，打开所有的 CSS 样式，如图 19.32 所示。

（6）双击选择符，可以重新打开“CSS 规则定义”对话框，修改原有的样式文件，最终列表属性

中的方框属性的参数如图 19.33 所示。



图 19.32　单击“全部”按钮后的面板



图 19.33　修改后的列表方框属性

（7）将列表内容样式中的字体更改为 13px，然后更改列表内容的补白属性，修改后列表内容的方框属性的参数如图 19.34 所示。



图 19.34　修改后的列表内容方框属性

（8）修改后的页面显示效果如图 19.35 所示。



图 19.35　修改列表属性后的显示效果

19.4　制作首页的主体部分

首页的主体部分可以分为两个部分，分别是左侧包含公告的侧栏部分，右侧含有新闻的内容部分。下面分别讲解它们的制作过程。

19.4.1　分析主体部分效果图

在制作之前，同样先要分析一下效果图，分清页面中的内容和修饰部分。主体部分的效果图如图 19.36 所示。



图 19.36　主体部分的效果图

从图 19.36 可以看出，左侧内容分为 3 个部分，分别为公告部分、热点推荐部分、业务咨询部分。右侧分为 4 个部分，分别是关于我们、今日新闻、点拨和时评、合作伙伴部分。下面分别进行制作。

19.4.2　制作主体部分的父元素

主体部分的父元素主要定义元素的居中和背景。

（1）选择“插入”|“布局元素”|“Div 标签”命令，添加新的布局元素。其中参数如图 19.37 所示。

（2）单击“新建 CSS 样式”按钮，定义 menu 的样式，其具体的参数如图 19.38、图 19.39 所示。



图 19.37　添加 main 元素

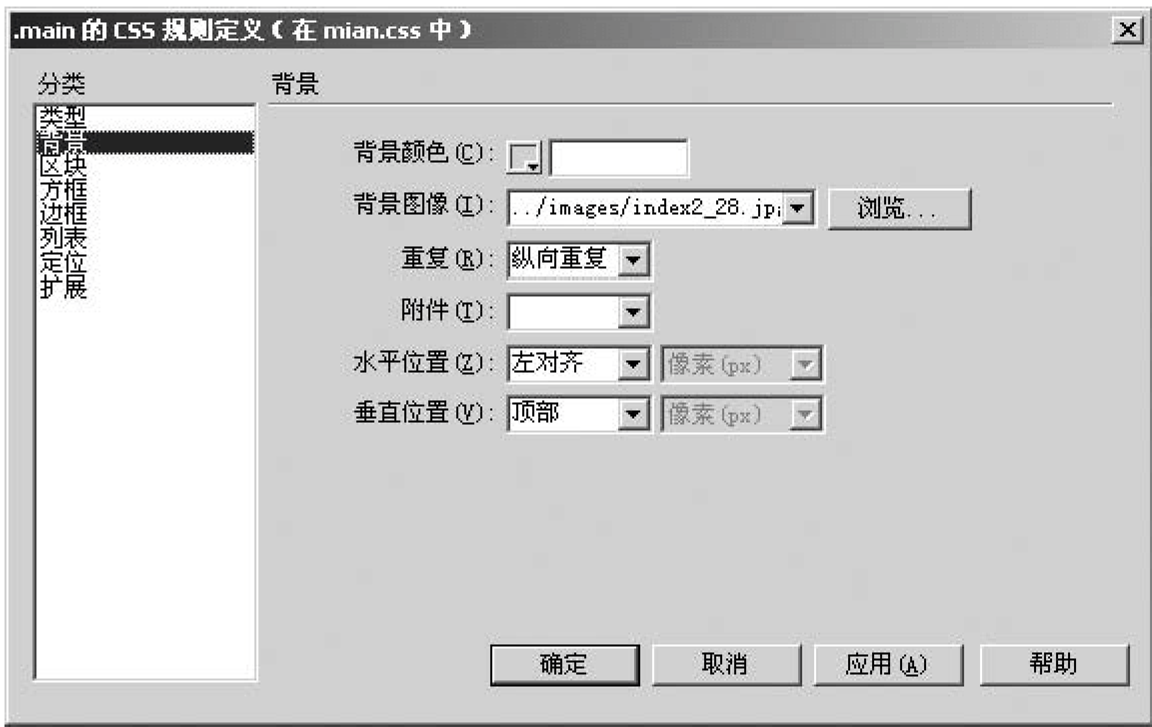


图 19.38　定义 main 元素的背景属性



19.39　定义 main 元素的方框属性

该样式使用边界属性定义了元素的居中，使用重复属性和背景图片定义了页面背景。其原理和前面章节中使用的原理完全相同。

19.4.3　制作主体左侧部分的样式

主体左侧部分分为 3 个部分来制作。

1．left 元素和公告部分的制作

left 元素是控制整个左侧内容的位置、宽度和高度的元素。

（1）选择“插入”|“布局元素”|“Div 标签”命令，添加新的布局元素。其中，参数中定义类名为 left。添加 left 元素。

（2）然后单击“新建 CSS 样式”按钮定义 left 的样式，其具体的参数如图 19.40 所示。

（3）制作公告部分，公告部分包括两个方面，即标题和公告内容，首先制作公告标题部分。

（4）同样先插入一个 div 元素，并添加类名为 notice_title。然后单击“新建 CSS 样式”按钮定义公告标题的样式，其具体的参数如图 19.41、图 19.42 所示。



图 19.40　left 元素的方框属性



图 19.41　公告标题的字体样式



图 19.42　公告标题的方框样式

（5）定义完公告标题样式后，添加公告标题文本“站内公告”。

（6）最后制作公告内容部分，添加一个新的 div 元素，定义类名为 notice-content。然后单击“新建 CSS 样式”按钮定义公告内容的样式，其具体的参数如图 19.43、图 19.44 所示。



图 19.43　公告内容的文本属性



图 19.44　公告内容的方框属性

定义完 left 元素和公告样式后，页面的显示效果如图 19.45 所示。

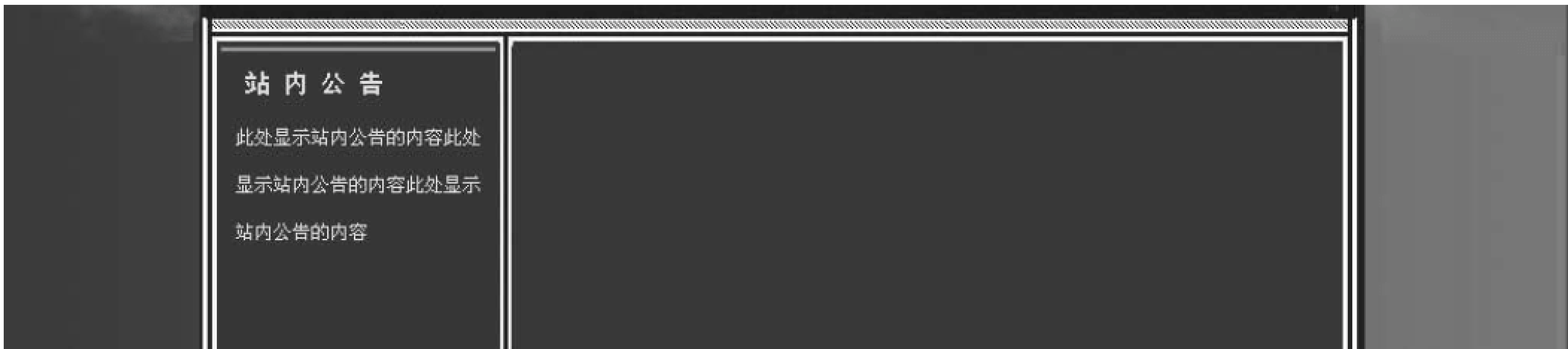


图 19.45　定义了 left 元素和公告样式后的效果

2．制作热点推荐部分

热点推荐部分，由 3 个结构样式相同的部分组成，下面以其中的一个为例，讲解制作方法。

（1）在公告的后面添加一个 div 元素，添加类名为 hot。然后单击“新建 CSS 样式”按钮，定义公告标题的样式，其具体的参数如图 19.46 所示。

（2）添加完 hot 元素后，在元素中添加图片元素，选择添加的图片，右击添加样式，如图 19.47、图 19.48 所示。



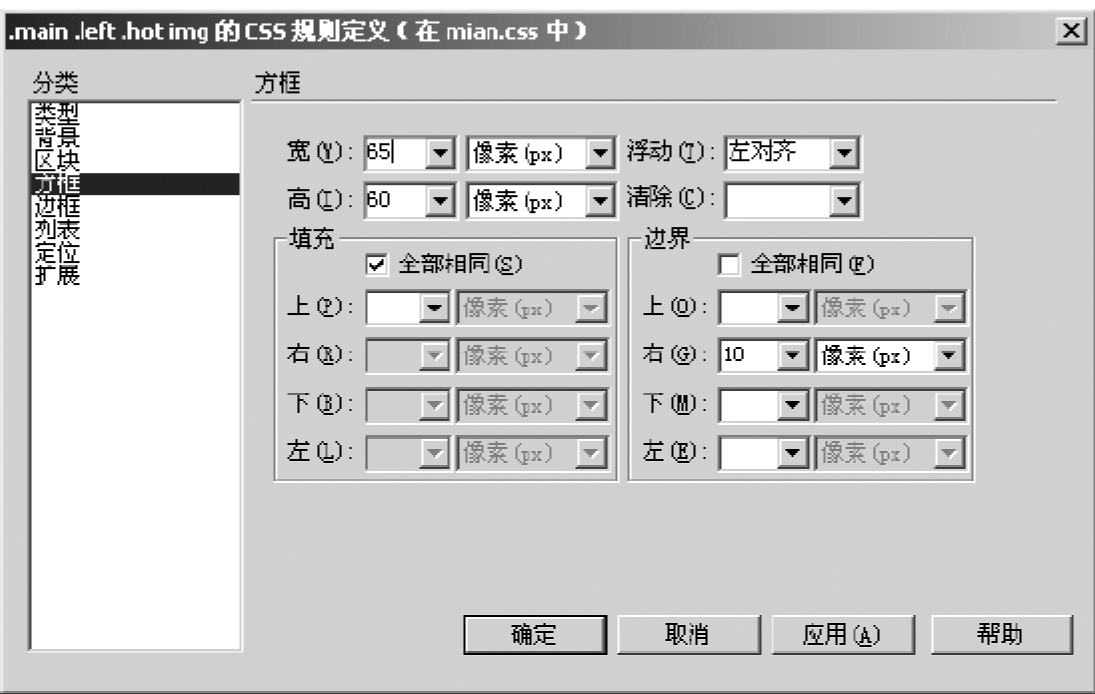


图 19.46　hot 元素的方框属性　　　　图 19.47　图片的方框属性

（3）添加热点的标题和内容。添加标题“房地产”，同时给标题添加空的超链接。在拆分视图中选择 a 和其中的内容，右击添加样式，使用默认值，如图 19.49 所示。



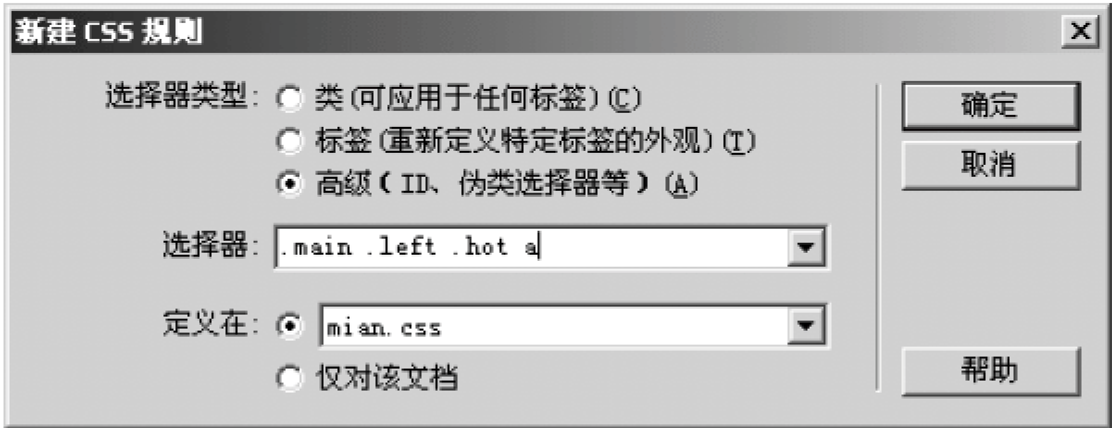


图 19.48　图片的边框属性　　　　图 19.49　添加标题的链接样式

（4）添加的链接样式如图 19.50 所示。在拆分视图的代码中后面添加换行符号
，然后添

加热点的内容。此时，热点部分的显示效果如图 19.51 所示。



图 19.50　标题的文本属性



图 19.51　热点部分显示的效果

（5）从图 19.51 可以看出，此时主要的问题是行高的问题，所以要修改 hot 的样式并添加行高属性，如图 19.52 所示。

（6）在拆分视图的代码窗口中，将 hot 元素和其包含的元素进行“复制”、“粘贴”，制作另外两个样式和结构相同的内容。然后修改图片和热点标题，最终热点部分的显示效果如图 19.53 所示。



图 19.52　添加行高属性



图 19.53　热点部分显示的效果

3．制作咨询热线部分

咨询热线部分很简单，只需要添加一个 div 元素，同时定义好行高，在内容中将标题和联系电话用换行符号分隔成两行即可。定义新添加的 div 元素，类名为 contact，并设置其样式如图 19.54、图 19.55 所示。



图 19.54　contact 元素的行高属性



图 19.55　contact 元素的方框属性

这样，左侧的内容就制作完成了。

19.4.4　制作主体右侧内容中关于我们的部分

在制作右侧的具体内容之前，首先要制作控制所有内容显示位置的父元素 right。

1. 制作父元素 right

调整光标到 left 元素结束符的后面，添加新的元素，定义类名为 right，同时定义样式，其参数如图 19.56 所示。

2. 制作关于我们的部分

（1）添加一个控制关于我们的元素，用来控制所有关于我们内容的位置。

（2）添加一个新的 div 元素，定义其类名为 aboutus，并设置其样式参数如图 19.57 所示。



图 19.56　right 元素的方框属性



图 19.57　aboutus 元素的方框样式

（3）在 aboutus 元素中添加新的 div 元素，定义类名为 content_title，并设置其样式参数如图 19.58、图 19.59、图 19.60 所示。



图 19.58　内容标题的文本属性



图 19.59　内容标题的背景属性

（4）设置完内容标题的样式后，在内容标题中添加一个类名为 title 的元素，同时在 title 中添加标题文本“关于我们”（因为是中文网站，所以用“关于我们”替代了效果图中的 About Us），此时内容标题的显示效果如图 19.61 所示。

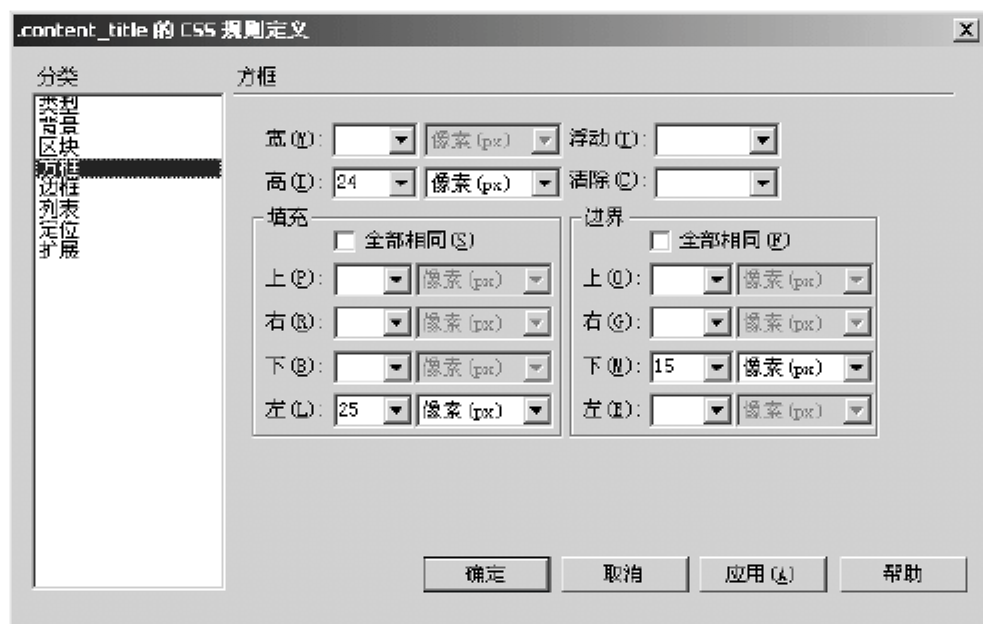
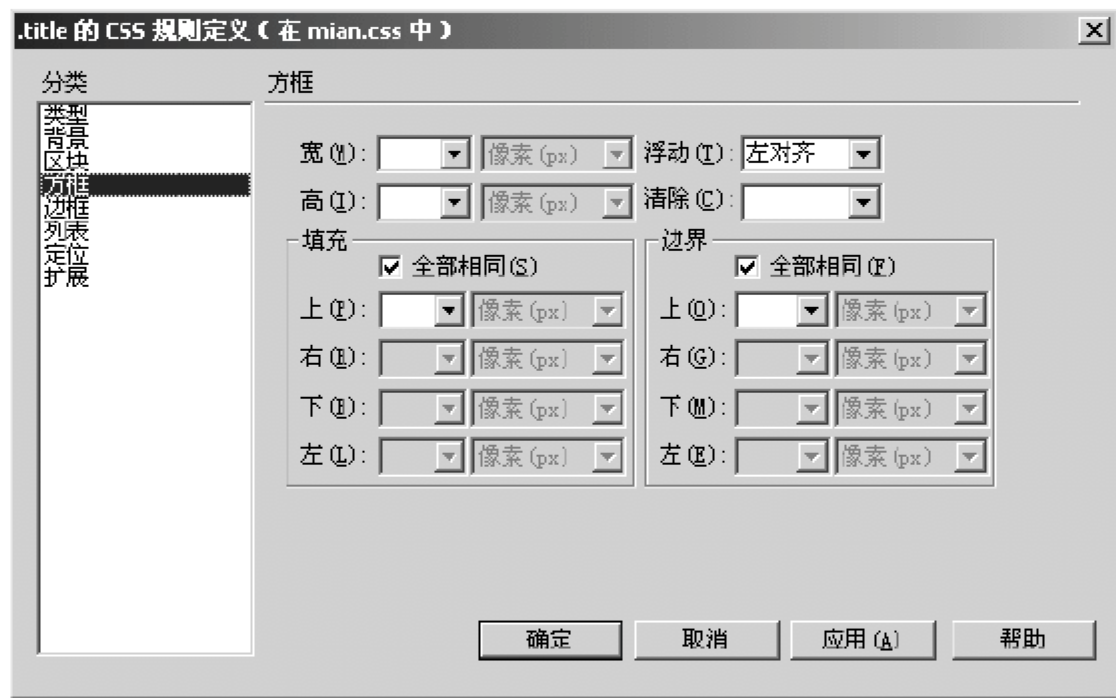




图 19.60　内容标题的方框属性

图 19.61　内容标题的显示效果

（5）之后再添加另一个类名为 more 的元素，同时在 more 元素中添加含有链接的文本 more，并且定义 title 元素的样式，如图 19.62 所示。设置 more 元素的属性如图 19.63 所示。



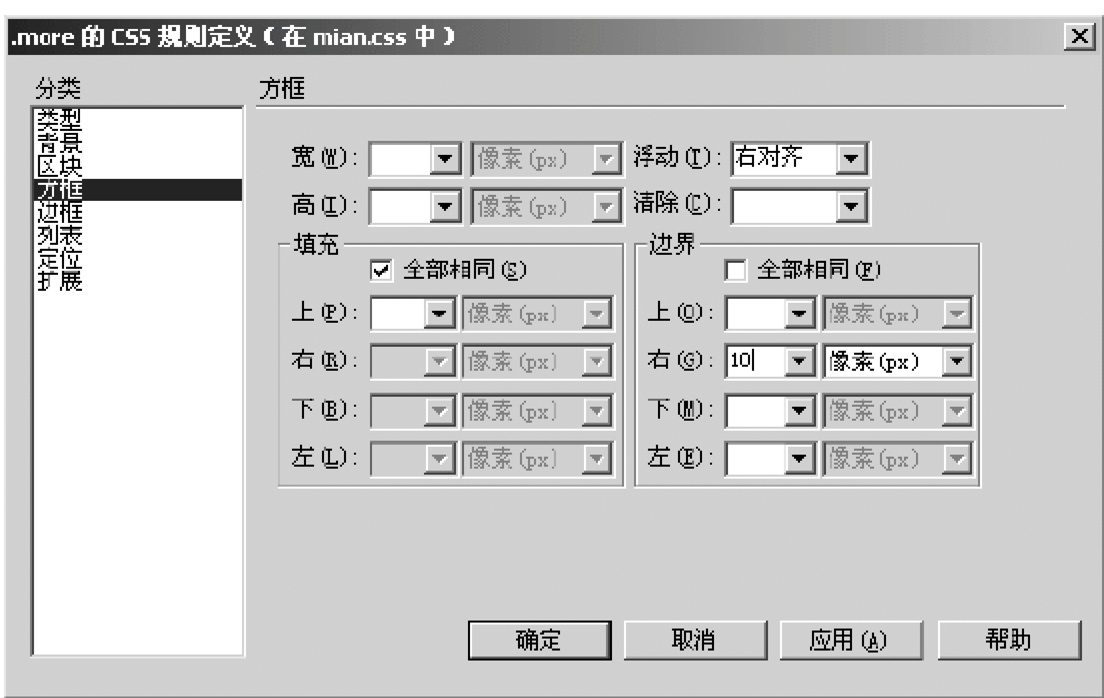


图 19.62　title 元素的方框属性

图 19.63　more 元素的方框属性

（6）同时定义 more 中的链接文本样式，如图 19.64 所示。

（7）因为使用了浮动属性，所以还要添加一个清除浮动的元素，其中样式参数如图 19.65、图 19.66 所示。

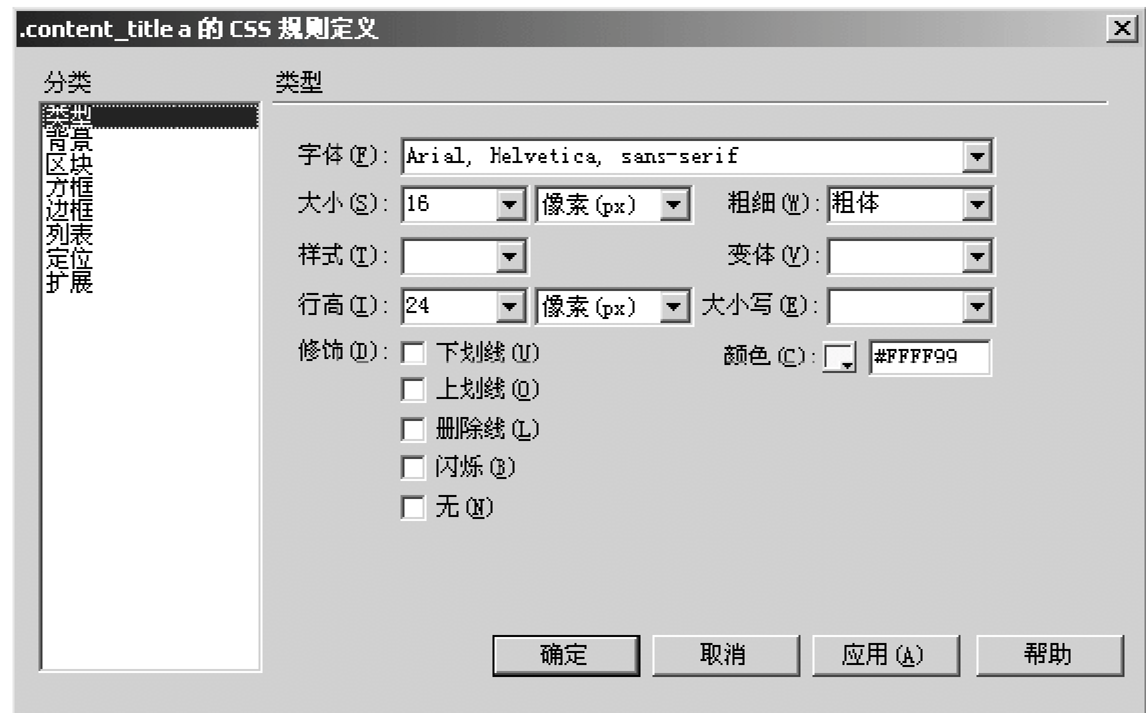




图 19.64　more 元素中链接的文本样式

图 19.65　清除浮动元素的行高属性

（8）添加展示图片和关于我们的内容，同时选择添加的图片并右击，添加 CSS 样式，具体参数如图 19.67、图 19.68 所示。



图 19.66　清除浮动元素的方框属性



图 19.67　展示图片的方框属性

此时，关于我们部分的显示效果如图 19.69 所示。



图 19.68　展示图片的边框属性



图 19.69　关于我们的显示效果

可以看出此时存在的问题是行高的问题，所以添加 aboutus 的行高属性，其参数如图 19.70 所示。



图 19.70　定义 aboutus 的行高属性

19.4.5　制作今日新闻部分

（1）同样先添加新的元素，定义类名为 news，并设置其参数如图 19.71、图 19.72 所示。



图 19.71　设置 news 元素的方框属性



图 19.72　设置 news 元素的边框属性

（2）news 部分的标题可以使用关于我们部分定义的样式，所以可以在拆分视图的代码窗口中直接“复制”、“粘贴”关于我们的相关代码，然后更改其内容，此时，新闻标题显示效果如图 19.73 所示。

（3）从图 19.73 可以看出，此时右侧文本 more 的链接样式并没有实现，其原因是因为，在关于我们的代码中用子选择符的方法限定了 more 的位置。在样式表控制面板中选择 main.css 选项，右击“转到代码”命令，转到 main.css 代码页面，将如下所示的代码：

```
.main .right .aboutus .content_title a
```

更改为下面的代码：

```
.content_title a
```

其中，选择符中定义的样式不变。更改后页面显示效果如图 19.74 所示。

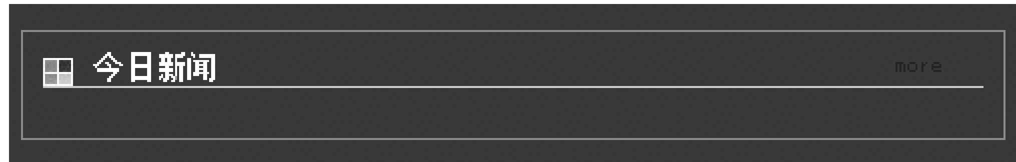


图 19.73　今日新闻部分的显示效果

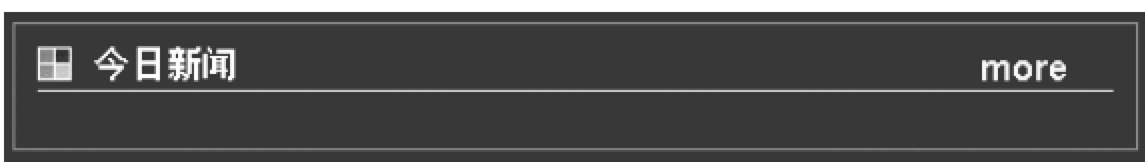


图 19.74　更改选择符后的显示效果

（4）制作新闻列表部分，使用添加导航列表相同的方法添加列表的内容。此时，界面会自动转换到拆分视图，在代码窗口中选择 ul 元素。定义类名为 newsnav，同时定义其样式如图 19.75、图 19.76 所示。



图 19.75　新闻列表的方框属性



图 19.76　新闻列表的列表属性

（5）选择其中的列表内容，右击并选择“CSS 样式”|“新建”命令，打开“新建 CSS 规则”对话框，更改默认选择符为如图 19.77 所示。

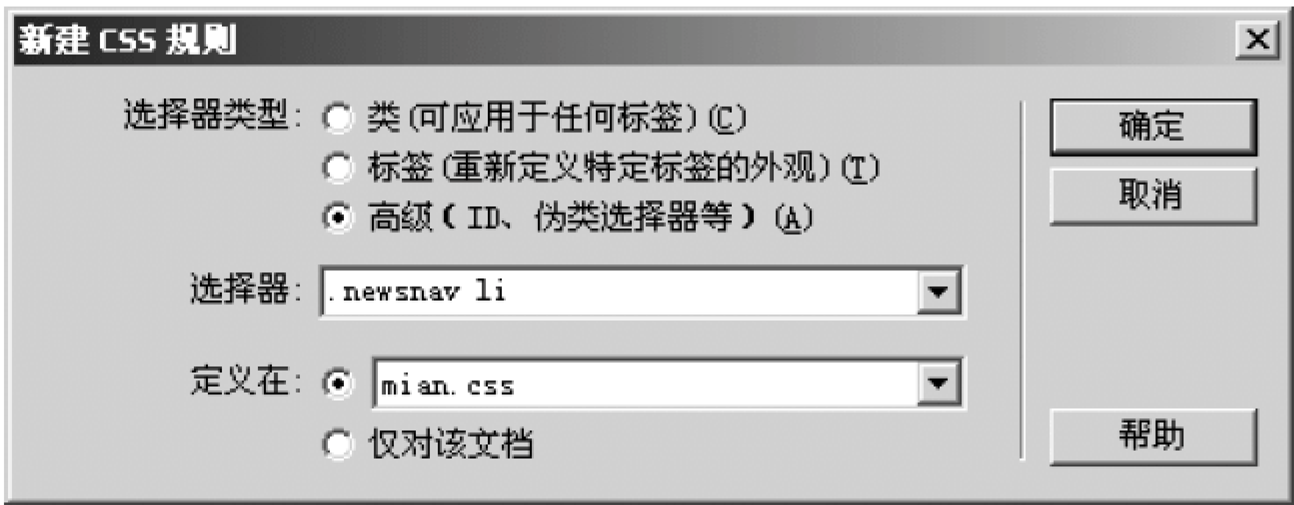


图 19.77　定义新闻列表的类名

（6）定义新闻列表内容的样式，如图 19.78、图 19.79、图 19.80 所示。

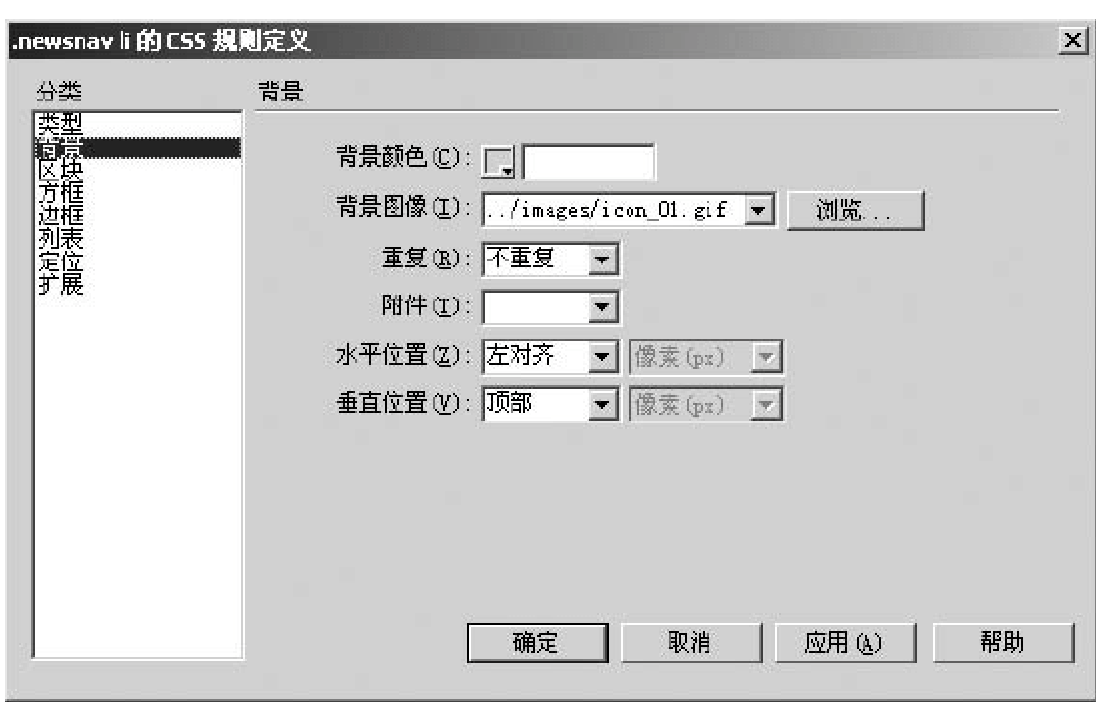


图 19.78　新闻列表内容的背景属性



图 19.79　新闻列表内容的方框属性

（7）定义好新闻列表属性后，页面新闻部分的显示效果如图 19.81 所示。



图 19.80　新闻列表内容的文本属性

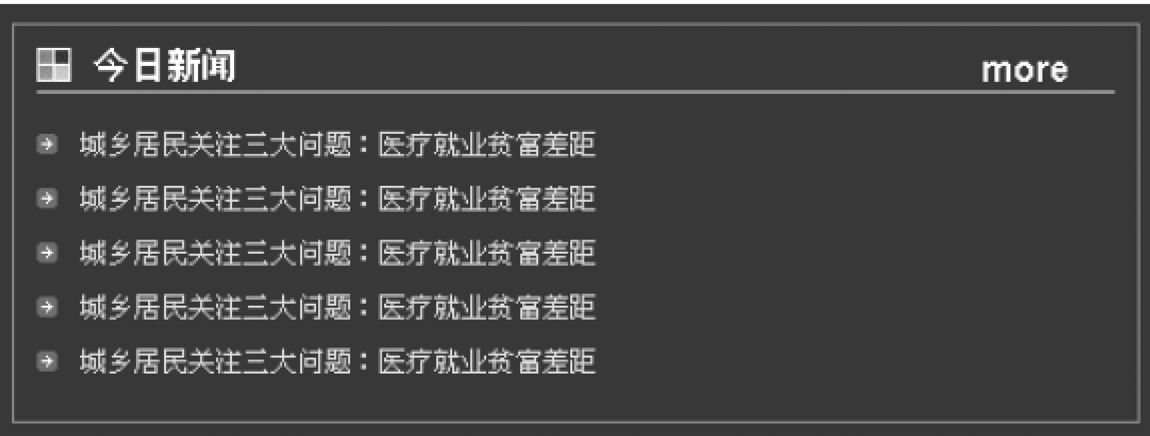


图 19.81　新闻列表的显示效果

19.4.6　制作点拨和时评的部分

点拨和时评部分的样式基本相同，区别在于位置不同。可以使用两个浮动的元素分别控制两个部分的位置。下面分别进行制作。

1．点拨部分的制作

（1）添加新的元素，并定义类名为 urged，然后定义元素的样式如图 19.82、图 19.83 所示。



图 19.82 urged 的方框属性



图 19.83 urged 的边框属性

（2）标题部分依然可以使用“复制”、“粘贴”的方法，统一使用 content_title 部分的结构和样式。这里就不再讲解了。

（3）内容列表的制作方法和新闻部分的列表制作方法类似，其中的区别在于，背景和补白属性不同，首先定义点拨列表 ul 的属性。选择 ul 和其中的内容，右击进入“新建 CSS 规则“对话框，更改默认的参数如图 19.84 所示。

注意：这样更改的主要原因是，因为时评部分将使用相同的列表样式，如果不取消 urged 类，则时评部分的列表不能继承现在定义的列表属性。

其具体的参数如图 19.85、图 19.86 所示。

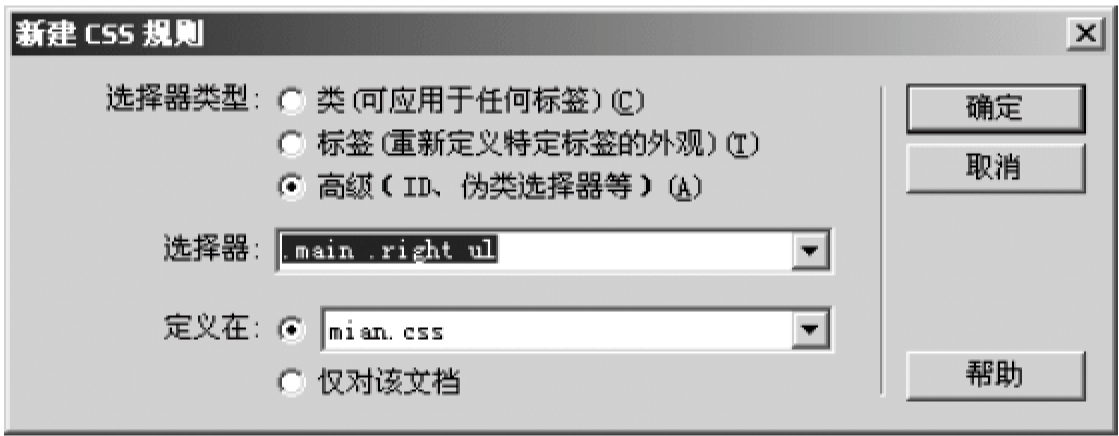


图 19.84 列表的默认参数



图 19.85 列表的方框属性

（4）定义 li 的属性，其具体参数如图 19.87 所示。

（5）定义完标题、列表属性后的页面显示效果如图 19.88 所示。



图 19.86　列表的列表属性

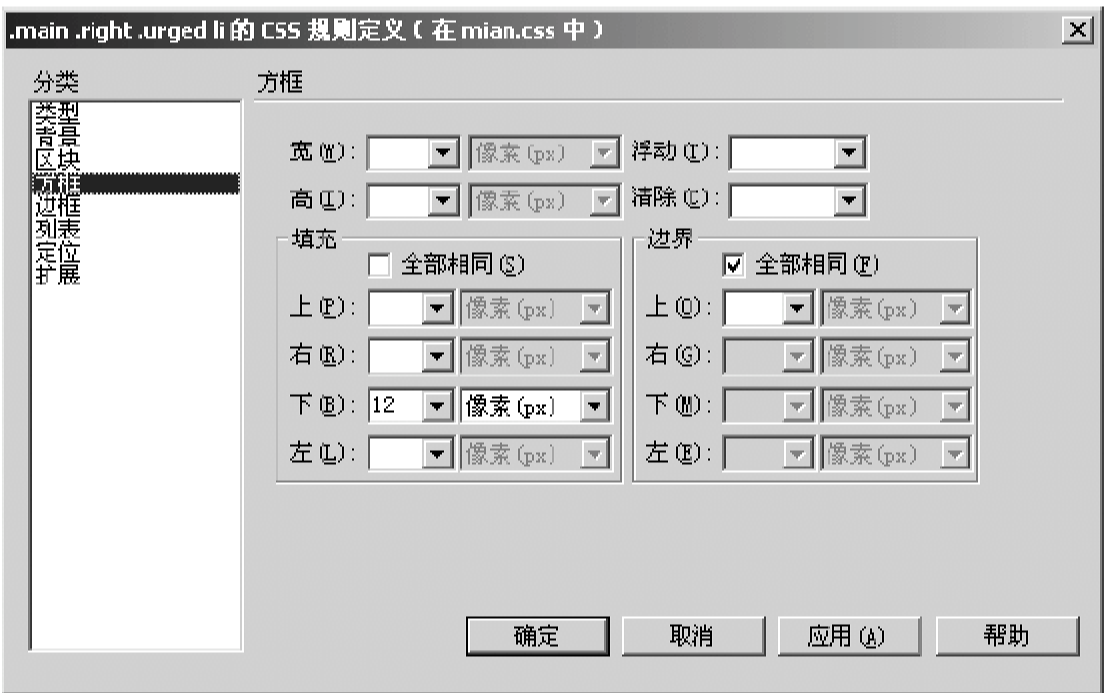


图 19.87　列表内容的方框属性

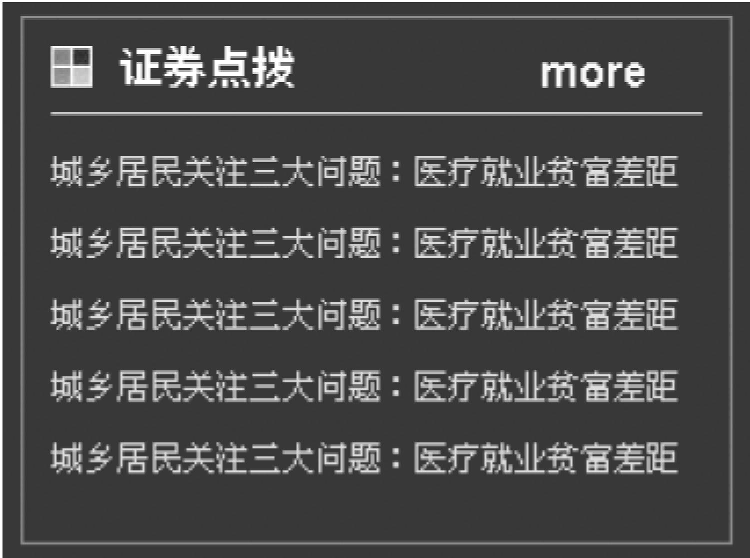


图 19.88　点拨部分的显示效果

2. 制作时评部分

（1）制作时评部分浮动的父元素。添加新的元素，定义其类名为 comment。设置其样式参数如图 19.89 所示。

（2）将点拨部分的内容和结构“复制”、“粘贴”到 comment 元素中，更改相关内容，显示效果如图 19.90 所示。



图 19.89　comment 元素的方框属性



图 19.90　点拨和时评部分的显示效果

（3）同样因为使用了浮动元素，所以还要使用清除浮动的元素。添加新的元素，在“插入 Div 标签”对话框的类下拉子菜单中选择 clear 类，制作好清除浮动元素。

19.4.7 制作合作伙伴部分

合作伙伴部分的制作分为以下几个部分。

1. 制作合作伙伴部分的父元素

添加新的 div 元素，定义类名为 partnership，并定义其样式如图 19.91、图 19.92 所示。

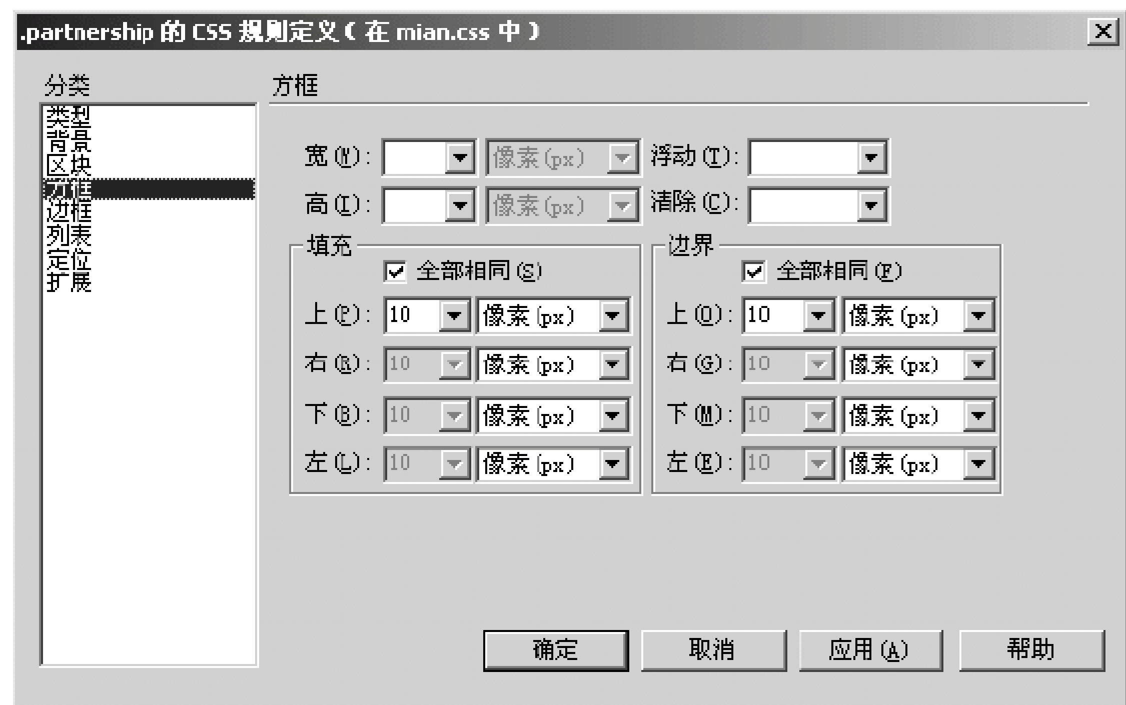


图 19.91 partnership 元素的方框属性

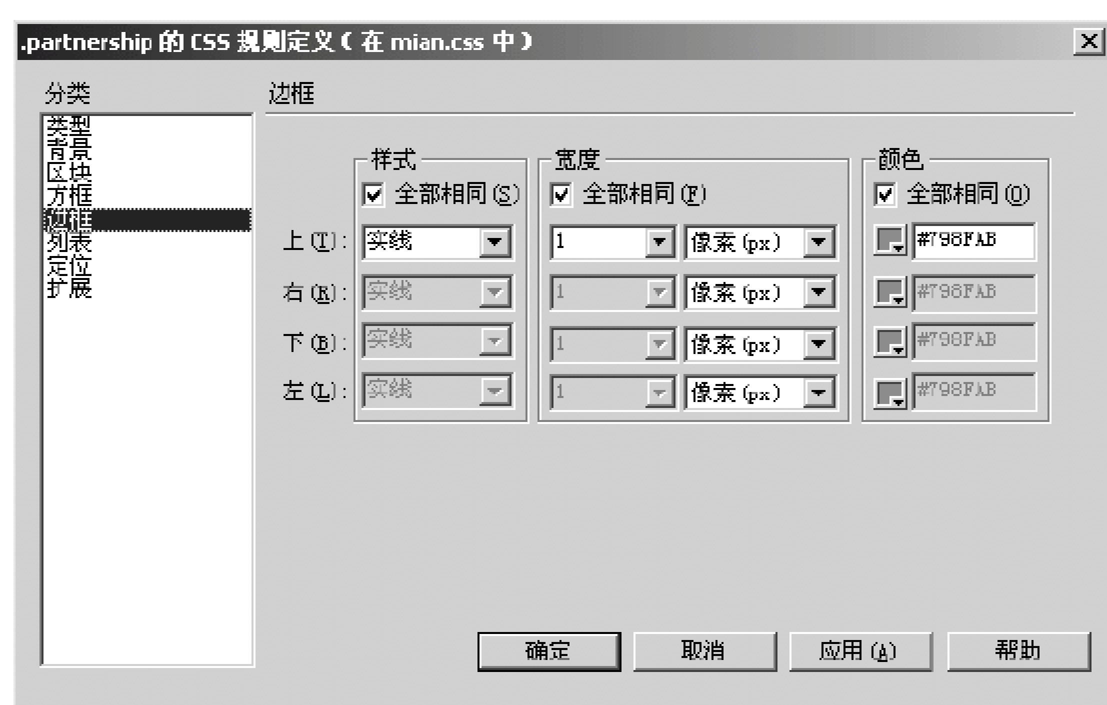


图 19.92 partnership 元素的边框属性

因为 partnership 元素中含有文本，所以还要定义行高属性，其参数如图 19.93 所示。

2. 内容的制作

（1）添加标题图片，并定义其浮动属性为 left，右边界属性为 20px。

（2）然后添加合作伙伴的内容部分，并使用换行符
进行分隔。此时，页面的显示效果如图 19.94 所示。

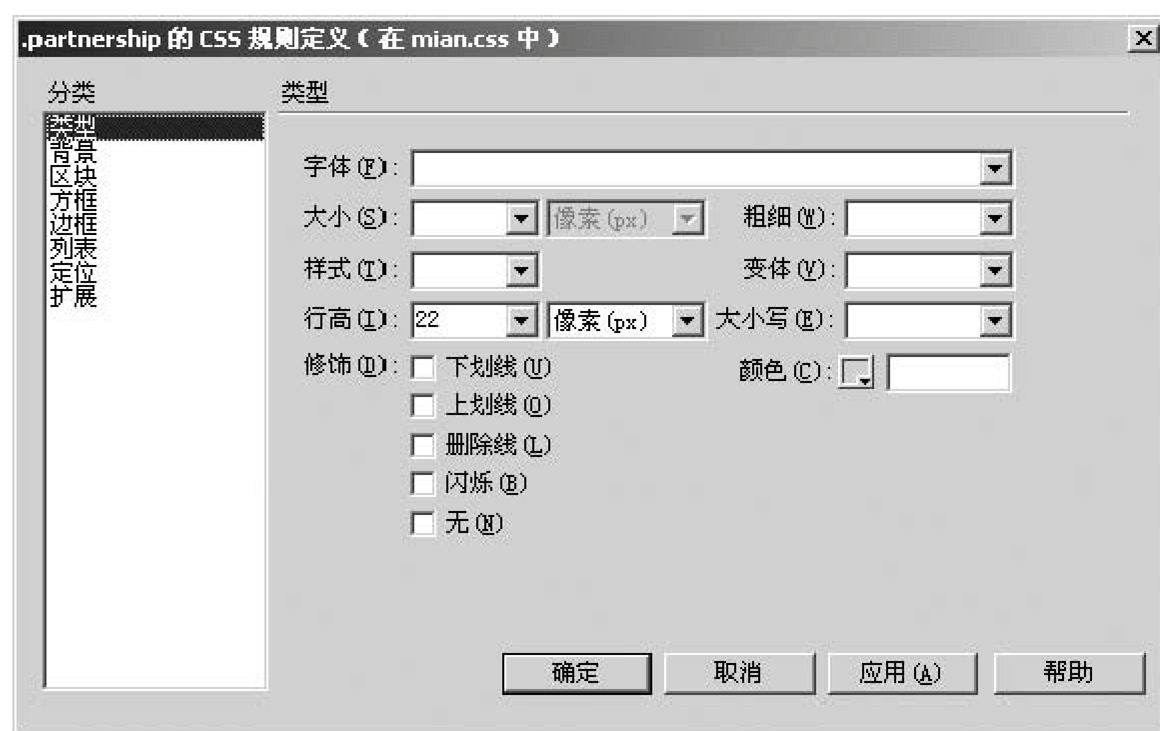


图 19.93 定义 partnership 元素中的文本属性

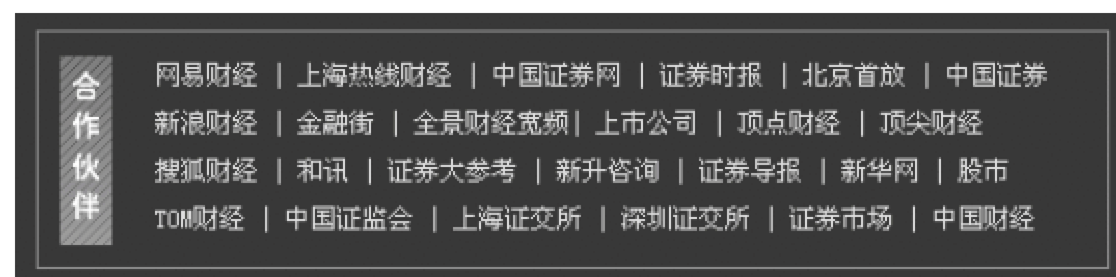


图 19.94 合作伙伴的显示效果

（3）从图 19.94 可以看出，此时的主要问题是，每行的开头部分的链接颜色没有改变。所以要重新定义这个部分的链接样式。选择每行开头的链接和内容，然后添加新的样式，定义类名为 partnership_head，同时设置样式如图 19.95 所示。

选择其他的内容，应用相同的样式，制作好每行开头内容的链接颜色。

因为左右两侧的 left 和 right 元素也使用了浮动属性，所以还要添加一个相应的清除浮动元素。



图 19.95　定义新的链接样式

19.5　制作首页的底部

首页的底部相对来说比较简单一些，主要由背景和居中的内容组成，其效果如图 19.96 所示。



图 19.96　底部的效果图

（1）制作底部的父元素，定义类名为 footer，其样式如图 19.97～图 19.100 所示。



图 19.97　footer 元素的文本属性



图 19.98　footer 元素的背景属性

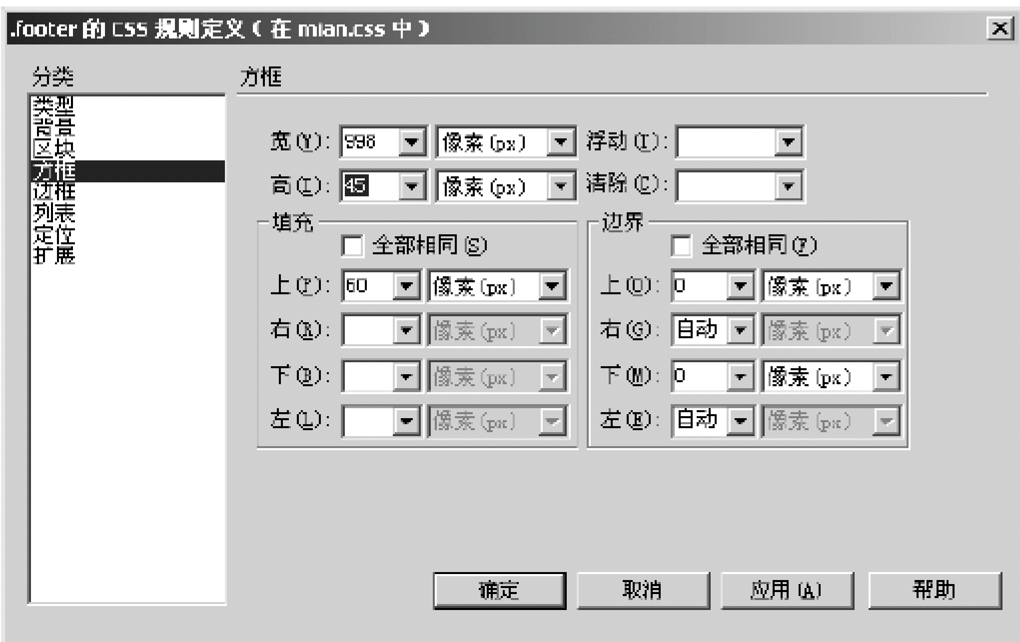


图 19.99　footer 元素的方框属性

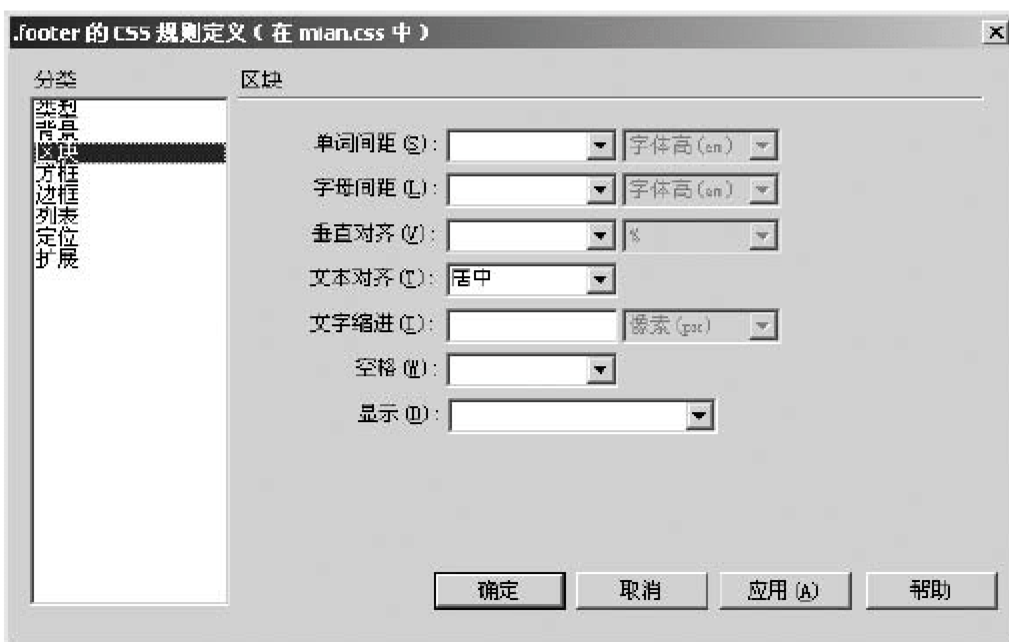


图 19.100　footer 元素的区块属性

（2）在 footer 元素中添加内容，首页底部就制作完成了。

19.6 首页的兼容问题

以上的制作过程都是在 IE 6.0 下进行的，制作好的首页在 Firefox2.0 中的显示效果如图 19.101 所示。



图 19.101　首页在 Firefox2.0 中的显示效果

从图 19.101 中可以看到，此时存在的显示问题出现在中间证券点拨和证券时评的部分。根据前面章节学习的知识可以知道，此时的问题应该是 IE 6.0 中浮动元素的双边界问题造成的。解决的办法有两个，一个是更改元素的间隔属性，即使用父元素的 padding 属性代替子元素的 margin 属性。另一个解决办法是使用!impirtant 声明。

【代码 19-3】下面使用!impirtant 声明，来解决现在的兼容问题。首先看一下此时相关的 CSS 样式代码，如程序 19.3 所示。

程序 19.3　相关的CSS样式代码

```
01   .urged {
02        padding: 10px;
03        width: 220px;
04        float:left;
05        border: 1px solid #798FAB;
06        margin: 5px;
07   }
08   .comment {
09        padding: 10px;
10        margin:5px;
11        float: right;
```

```
12          border: 1px solid #798FAB;
13          width: 220px;
14    }
```

【代码 19-4】此时要更改的部分是两个元素的 margin 属性，要使得在 IE 6.0 和 Firefox 浏览器中的边距相同，更改后的代码如程序 19.4 所示。

程序 19.4　相关的CSS样式代码

```
01    .urged {
02          padding: 10px;
03          width: 220px;
04          float:left;
05          border: 1px solid #798FAB;
06          margin-left:10px !important;
07          margin: 5px;
08    }
09    .comment {
10          padding: 10px;
11          margin-right:10px !important;
12          margin:5px;
13          float: right;
14          border: 1px solid #798FAB;
15          width: 220px;
16    }
```

【深入学习】程序 19.3 和程序 19.4 实现的是相同的功能，不过程序 19.4 更完整，其第 6 行和第 11 行使得 IE 6.0 中和 Firefox 浏览器中的边距相同。

19.7　二级页面的制作

从效果图中可以看出，首页和二级页面的头部、左侧、底部都是相同的，所以只需要更改首页右侧内容部分即可。二级页面的中间内容部分的效果图如图 19.102 所示。



图 19.102　二级页面的中间内容部分的效果图

从图 19.102 可以看出，此时右侧内容部分是一个新闻列表，其标题和新闻列表内容的样式都与首页的相同，所以可以使用首页的样式。

（1）将首页另存为 newsroom.html 页，注意更改页面标题。

（2）将首页右侧的无关内容删除，删除后的页面显示效果如图 19.103 所示。



图 19.103　删除右侧内容后的显示效果

（3）将“关于我们”修改成“今日新闻”，并添加相关的新闻列表。定义列表 ul 的样式为 newsnav，页面的显示效果如图 19.104 所示。

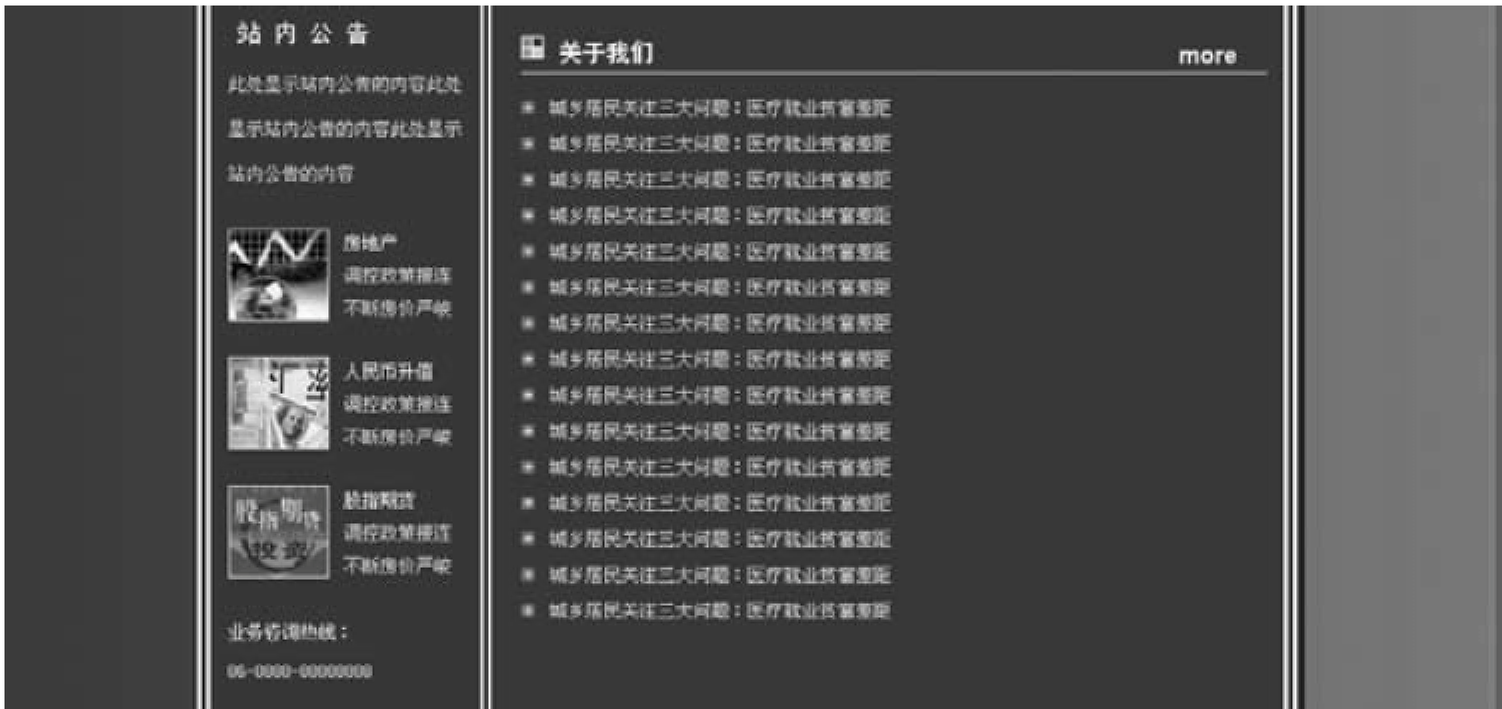


图 19.104　使用 newsnav 列表属性后的效果

（4）制作分页部分。同样先添加一个 div 元素，定义其类名为 page。定义其方框属性参数如图 19.105 所示。

（5）添加分页的内容，同时添加 select 表单。添加 select 表单的方法是，选择“插入”|“表单”|“列表/菜单”命令，添加表单。

（6）选择表单添加样式，定义类名为 select，其具体参数如图 19.106 所示。



图 19.105 定义 page 元素的方框属性



图 19.106 定义表单的样式

（7）定义完以上样式后，页面的显示效果如图 19.107 所示。

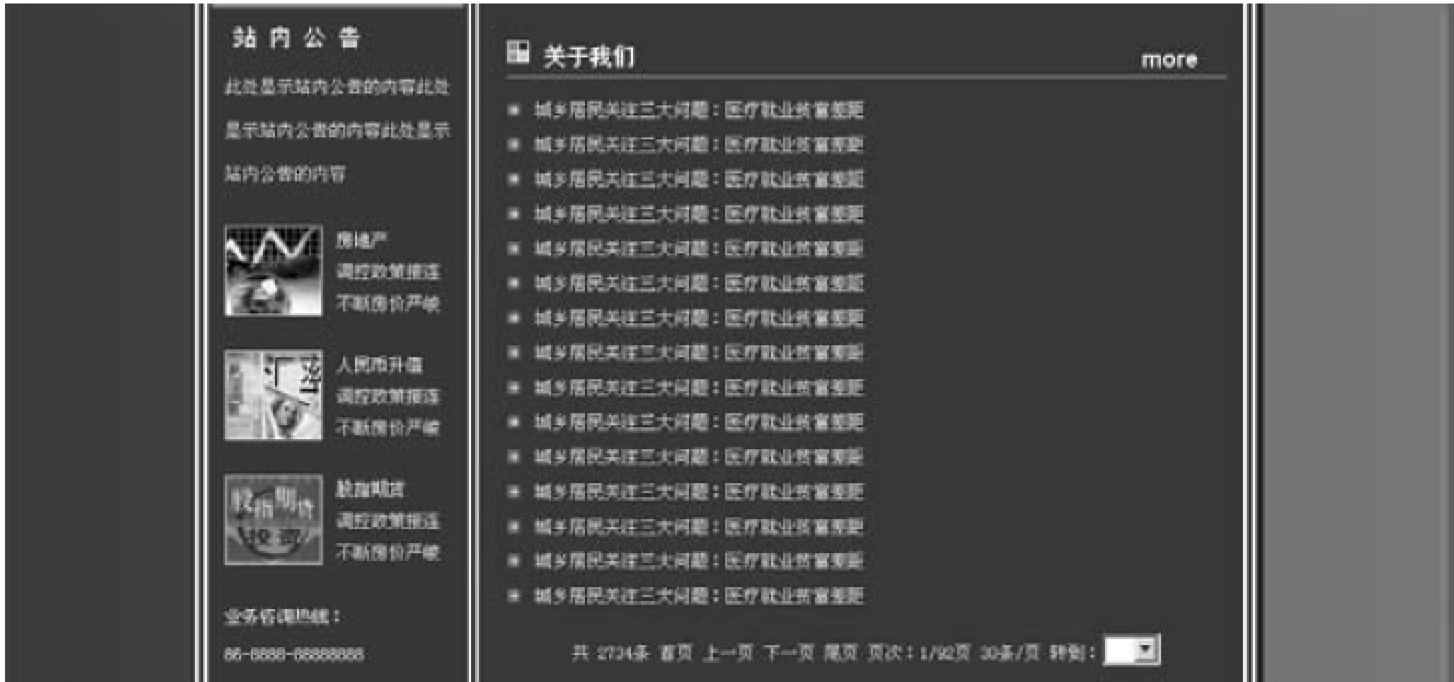


图 19.107 二级页面最终的显示效果

（8）将页面在 Firefox2.0 中测试，没有兼容问题。

19.8 小 结

如果说前面的实例全部手写代码让你很头疼，那本章的制作方式就有点太简单了。Dreamweaver 俗称网页剑客，是专门用来制作网页的工具。本章依据第 18 章的制作思路，通过 Dreamweaver 进行一些设置，可以自动生成网页所需要的 HTML 代码和 CSS 代码。学习完本章后，读者会发现，拥有一个开发工具可以更方便、更快捷地制作网站。

附录 A

A.1 HTML 4.0 快速参考

HTML 虽然比较简单，但它的标签名称和属性都是固定的，不像 XML 那样可以自己定义，本附录给出了 HTML 元素所具有的通用属性，供读者参考。

A.1.1 通用属性

通用属性	说明
ID	ID属性为文档中的元素指定了一个独一无二的身份标识，用于样式表和脚本引用。在定义ID属性时，必须注意此属性值由英文字母开头，后面可以跟任意字母（大写A～Z和小写a～z）、数字（0～9）、连字符（-）、下划线（_）、冒号（:）以及点号（.） 注意：ID属性与NAME属性使用相同的名称空间，因此不能在同一个文档中为ID和NAME属性定义相同的名称，以防止发生混乱
Class	Class属性定义了特定标记符的类，用于样式表和脚本引用。使用Class属性可以为标记符定义类别，此时可以说该标记符是属于该类别的。一个类别中也可以包含多个标记符
Style	Style属性用于为一个单独的标记符指定样式，也就是指定行内样式（inline style，也称为直插式样式）
Title	Title属性与TITLE标记符不同（TITLE标记符在文档中只能出现一次），它可以为文档中任意多个标记符指定参考标题信息。通常，浏览器将参考标题信息以即时提示（tooltip，也称为工具栏提示）的方式显示出来，以便浏览者查看

A.1.2 HTML 文档结构元素

语法	常用属性	说明
<HTML> </HTML>	无	开始标记符和结束标记符都可以省略。HTML标记符说明此文档是一个HTML文档
<HEAD> </HEAD>	无	开始标记符和结束标记符都可以省略。HEAD元素包含文档的头部信息，如标题、关键字、说明和样式表等。一般位于<HTML>标记之后，<BODY>标记之前。对于框架文档，位于<FRAMESET>标记之前
<BODY> </BODY>	❑ BACKGROUND=URL（文档的背景图像） ❑ BGCOLOR=Color（文档的背景色） ❑ TEXT=Color（文档中文本的颜色） ❑ LINK=Color（文档中链接的颜色） ❑ VLINK=Color（文档中已被访问过的链接的颜色）	开始标记符和结束标记符都可以省略。BODY元素中包含文档体，也就是文档的正文。 对于非框架文档，BODY位于HEAD之后； 对于框架文档，如果包含NOFRAMES标记符，则BODY必须位于该标记符内，否则不能包含BODY标记符

续表

语　法	常用属性	说　明
<BODY> </BODY>	❑ ALINK=Color（文档中活动链接的颜色） ❑ ONLOAD=Script（文档加载时执行脚本的事件） ❑ ONUNLOAD=Script（文档退出时执行脚本的事件） ❑ 通用属性	
<TITLE> </TITLE>	无	TITLE标记符位于HEAD标记符内，它包含的内容是文档的标题。每个文档在HEAD中有且仅有一个TITLE。TITLE标记符中包含的内容将在浏览器的标题栏中显示
<META>	❑ NAME=name（名字） ❑ HTTP-EQUIV=Name（HTTP 相应标题名） ❑ CONTENT=CDATA（相关数据）	META标记符中包含了网页的元数据信息，诸如文档关键字、作者信息等。文档的HEAD标记符内可以包含任意数量的<META>元素
<DIV> </DIV>	❑ ALIGN=[left | center | right | justify]（水平对齐方式） ❑ 通用属性	DIV标记符用于包含行内元素（也称为字符级元素或文本级元素）和块级元素，以便定义一个块。通常，该元素与Class和ID等属性联合使用，以便在样式表中为某一块内容定义样式。如果不使用样式表，DIV标记符常用于设置段落对齐。例如，<DIV align=center></DIV>可以为包含在其中的内容设置居中对齐（与<CENTER></CENTER>相同）
<SPAN> </SPAN>	通用属性	SPAN标记符与DIV标记符类似，但通常用于包含行内元素。例如，如果定义了以下样式：.red{color:red}，则可以使用以下语句为部分文本设置红色： <P>部分为<SPAN class="red">红色</SPAN>的文本</P>
<H1>...</H1> …… <H6>...</H6>	❑ ALIGN=[left | center | right | justify]（水平对齐方式） ❑ 通用属性	H1～H6元素用于定义从1级到6级标题，可以使用align属性设置标题的对齐方式
<ADDRESS> </ADDRESS>	通用属性	此标记符用于提供联系信息，通常用斜体字显示其中的内容

A.1.3　文本元素

语　法	常用属性	说　明
<ABBR> </ABBR>	通用属性	ABBR元素用来标记缩写，通常与title属性一起使用。例如，<ABBR title="Structured Query Language">SQL </ABBR>，则当浏览者将鼠标移动到SQL字样上时，将显示即时提示"Structured Query Language"

续表

语　法	常用属性	说　明
<ACRONYM> </ACRONYM>	通用属性	ACRONYM元素被用来标记首字母缩略词。与ABBR元素类似，它常常与title属性一起使用。例如，<ACRONYM title="Structured Query Language">SQL</ACRONYM>
<BLOCKQUOTE> </BLOCKQUOTE>	❑ CITE=URL（引用源） ❑ 通用属性	BLOCKQUOTE元素定义了一个块引用，其中可以包含块级元素（如P和TABLE）。表示<BLOCKQUOTE></BLOCKQUOTE>中包含的内容是引自cite属性所指定的源（例如，http://www.microsoft.com）
 	❑ CLEAR=[left | all | right |none]（清除浮动对象） ❑ 通用属性	 标记符用于强行中断当前行，多个 标记符可以创建多个空行。 标记符通常用于简单的格式设置
<CITE> </CITE>	通用属性	CITE元素用以标记引用内容，诸如杂志报纸的标题等。浏览器一般将<CITE></CITE>中的内容显示为斜体字
<CODE> </CODE>	通用属性	CODE元素用于标记文档中的代码，通常，浏览器将<CODE></CODE>中的内容显示为等宽字体
<DEL> </DEL>	❑ CITE=URL（包含删除原因信息的URL） ❑ DATETIME=Datetime（删除时间） ❑ 通用属性	DEL元素用来标记文档中已删除的内容，可以用title属性给出简单的删除原因。通常，浏览器将包含在<DEL></DEL>中的文字添加上删除线。为确保在多数浏览器中都可以有删除线效果，也可以结合使用STRIKE或S元素。例如，以下语句可以确保在多数浏览器上显示下划线效果：<DEL cite="www.mysite.com/book"title=" Sold out"> <H3><S>《HTML4教程》</S></H3></DEL>
<DFN>...</DFN>	通用属性	DFN元素用于指定一个定义，在浏览器中通常用斜体字显示
<EM> </EM>	通用属性	EM元素用于对其中包含的内容进行强调，通常浏览器用斜体字显示<EM></EM>中包含的内容。也可以使用样式表为其指定特殊效果
<HR>	❑ ALIGN=[left | center | right]（指定水平对齐方式） ❑ NOSHADE（实线） ❑ SIZE=Pixels（线宽） ❑ WIDTH=Length（线长） ❑ 通用属性	HR元素用于在网页中添加一条水平线

续表

语　法	常 用 属 性	说　明
<INS> </INS>	❑ CITE=URL（说明插入原因信息所在的URL） ❑ DATETIME=Datetime（插入时间） ❑ 通用属性	INS元素用于包含被插入的内容。在IE 5中以下划线显示包含在<INS></INS>中的文字。用户也可以自定义样式表，以便指定特定的显示格式
<KBD>...</KBD>	通用属性	KBD元素用于包含键盘录入的文字，在浏览器中通常以等宽字体显示
<P> </P>	❑ ALIGN=[left | center | right |justify]（设置水平对齐方式） ❑ 通用属性	结束标记符可以省略，但使用样式表时要使用结束标记符。P元素用于在网页中分段
<PRE> </PRE>	❑ WIDTH=Number（宽度） ❑ 通用属性	PRE元素用于包含预先格式化的文本。也就是说，包含在<PRE></PRE>中的内容将以所设置的格式显示
<Q> </Q>	❑ CITE=URL（引用源） ❑ 通用属性	Q元素用于表示短的行内引用。如果需要表示更长的引用，应使用BLOCKQUOTE元素。由于一般的浏览器并不支持此元素，因此需要用样式表指定该元素的格式
<SAMP> </SAMP>	通用属性	SAMP元素标记了网页中的输出样本，如程序的输出。通常，浏览器将<SAMP></SAMP>中的文字以等宽字体显示。用户也可以用样式表自定义该元素的样式
<STRONG> </STRONG>	通用属性	STRONG元素用于对包含在其中的内容进行强调，浏览器通常用粗体字显示包含在<STRONG></STRONG>中的内容。用户也可以用样式表来规定显示样式
_{...}	通用属性	SUB元素用于定义下标
^{...}	通用属性	SUP元素用于定义上标
<VAR> </VAR>	通用属性	VAR元素用于标记变量或程序参数。浏览器通常用斜体字显示包含在<VAR></VAR>中的文字。也可以用样式表自定义该元素的样式

A.1.4　字体样式元素

语　法	常 用 属 性	说　明
<B>...</B>	通用属性	B元素可以使文本以粗体形式出现
<BASEFONT>	❑ SIZE=CDATA（指定默认字体大小，范围为 1～7，默认值是 3） ❑ COLOR=Color（指定默认字体颜色） ❑ FACE=CDATA（指定默认字体） ❑ ID=ID（唯一的 ID）	BASEFONT元素允许作者规定基本字体的大小、颜色和“字体”。但由于样式表的出现，在HTML 4中它是已过时的用法
<BIG>...</BIG>	❑ 通用属性	BIG元素规定文本以大字体显示

续表

语　　法	常 用 属 性	说　　明
<FONT> </FONT>	❑ SIZE=CDATA（字体大小调整） ❑ COLOR=Color（字体颜色调整） ❑ FACE=CDATA（字体样式调整） ❑ 通用属性	FONT元素用于设置所包含字体的大小、颜色和“字体”。由于样式表单的出现，FONT元素在HTML 4中属已过时的用法
<I>...</I>	❑ 通用属性	I元素规定文本以斜体显示
<S>...</S>	❑ 通用属性	S元素规定文本以包含删除线的方式显示，效果与STRIKE元素相同
<SMALL> </SMALL>	❑ 通用属性	SMALL元素规定文本以小字体显示
<STRIKE> </STRIKE>	❑ 通用属性	STRIKE元素规定文本显示时加删除线，效果与S元素相同
<TT>...</TT>	❑ 通用属性	TT元素规定文本以电报文字体或等宽字体显示
<U>...</U>	❑ 通用属性	U元素规定文本显示时加下划线

A.1.5　列表元素

<table>
<tr><th>语　　法</th><th>常 用 属 性</th><th>说　　明</th></tr>
<tr><td><UL>
</UL></td><td>❑ TYPE=[disc ｜ square ｜ circle]（编号样式）
❑ COMPACT（紧凑显示）
❑ 通用属性</td><td>UL元素定义了一个无序列表，其中包含一个或多个LI元素来定义实际的列表项</td></tr>
<tr><td><OL>
</OL></td><td>❑ TYPE=[l | a | A | i | I]（编号方式）
❑ START=Number（起始数）</td><td>OL元素定义了一个有序列表。OL元素中包含一个或多个LI元素来定义实际的列表项。与无序列表不同，列表项有一个明确的顺序</td></tr>
<tr><td></td><td>❑ COMPACT（紧凑显示）
❑ 通用属性</td><td>表项由浏览器自动编号</td></tr>
<tr><td><LI>
</LI></td><td>❑ TYPE=[disc | square | circle | l | a | A | i | I]（列表项标记样式）
❑ VALUE=Number（序列号）
❑ 通用属性</td><td>结束标记可以省略，但使用样式表时应使用结束标记。LI元素定义了一个列表项</td></tr>
<tr><td><DL>
</DL></td><td>❑ COMPACT（紧凑显示）
❑ 通用属性</td><td>DL元素定义了一个定义列表。定义列表中的条目是通过使用DT元素和DD元素创建的。DT元素给出了术语名，而DD元素给出了术语的定义</td></tr>
<tr><td><DT>
</DT></td><td>❑ 通用属性</td><td>结束标记可以省略，但使用样式表时应使用结束标记。DT元素在定义列表中定义了一个术语</td></tr>
<tr><td><DD>
</DD></td><td>❑ 通用属性</td><td>结束标记可以省略，但使用样式表时应使用结束标记。DD元素在定义列表中为一个术语提供定义数据</td></tr>
</table>

续表

语　　法	常 用 属 性	说　　明
<DIR> </DIR>	❑ COMPACT（紧凑显示） ❑ 通用属性	DIR元素定义了一个目录列表，其中包含一个或多个定义实际列表项的LI元素。此时，LI元素中不可包含块级元素。在HTML 4.0中，DIR元素已被UL元素取代
<MENU> </MENU>	❑ COMPACT（紧凑显示） ❑ 通用属性	MENU元素定义了一个菜单列表，其中包含一个或多个LI元素来定义实际菜单项。此时，LI元素中不应包含块级元素。MENU元素在HTML 4.0中属过时的用法

A.1.6　表格元素

语　　法	常 用 属 性	说　　明
<TABLE> </TABLE>	❑ SUMMARY=Text（表格说明） ❑ WIDTH=Length（表宽） ❑ BORDER=Pixels（边框宽度） ❑ FRAME=[void | above | below | hsides | lhs | rhs | vsides | box | border]（外边框） ❑ RULES=[none | groups | rows | cols | all]（表格框线） ❑ CELLSPACING=Length（单元格间距） ❑ CELLPADDING=Length（单元格填充距）	TABLE元素用于定义表格，所有表格中的内容都应包含在<TABLE>和</TABLE>中
	❑ ALIGN=[left | center | right]（表格对齐） ❑ BGCOLOR=Color（表格背景色） ❑ 通用属性	
<CAPTION> </CAPTION>	❑ ALIGN=[top | bottom | left | right]（对齐方式） ❑ 通用属性	CAPTION元素定义了表格的标题，使用时CAPTION标记符必须放在表格最开头（即<TABLE>之后）
<THEAD> </THEAD>	❑ ALIGN=[left | center | right | justify | char]（组中单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（组中单元格的垂直对齐方式） ❑ 通用属性	THEAD元素定义了表格的表头，一个表格中最多可含有一个THEAD标记符。使用时，THEAD标记符必须跟在<CAPTION>、<COL>或<COLGROUP>后，在<TFOOT>和<TBODY>之前。目前，多数浏览器还不支持THEAD标记符
<TFOOT> </TFOOT>	❑ ALIGN=[left | center | right | justify | char]（组中单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（组中单元格的垂直对齐方式） ❑ 通用属性	TFOOT元素定义了表格的脚注行，一个表格中最多可含有一个TFOOT标记符。TFOOT标记符必须跟在THEAD后，在TBODY之前。目前多数浏览器还不支持TFOOT标记符

续表

语　法	常 用 属 性	说　明
<TBODY> </TBODY>	❑ ALIGN=[left | center | right | justify | char]（组中单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（组中单元格的垂直对齐方式） ❑ 通用属性	TBODY在表格中定义了一组数据行，表格中至少有一个TBODY标记符。TBODY必须跟在可选的TFOOT后。如果表格中仅有一个TBODY，且不含THEAD和TFOOT，则TBODY的起始和结尾标记可省略
<COLGROUP> </COLGROUP>	❑ SPAN=Number（组的列数） ❑ WIDTH=MultiLength（每列宽度） ❑ ALIGN=[left | center | right | justify | char]（组中单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（组中单元格的垂直对齐方式） ❑ 通用属性	COLGROUP元素定义了一个表格中的列组。使用列组时，COLGROUP元素必须放在可选的CAPTION元素之后，且在可选的THEAD元素之前
<COL>	❑ SPAN=Number（列数） ❑ WIDTH=MultiLength（列宽度） ❑ ALIGN=[left | center | right | justify | char]（列单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（列单元格的垂直对齐方式） ❑ 通用属性	COL元素定义了一个表格列的属性。如果使用此元素，则必须放在可选的CAPTION元素之后，且在可选的THEAD元素之前。与COLGROUP不同，COL并不在结构上将表格列分组，而是仅定义若干表格列所共享的属性。COL标记符也可以位于COLGROUP标记符之中，此时COL的属性将覆盖COLGROUP的属性
<TR> </TR>	❑ ALIGN=[left | center | right | justify | char]（组中单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（组中单元格的垂直对齐方式） ❑ BGCOLOR=Color（背景色） ❑ 通用属性	TR元素定义了一个表格行。TR必须出现在由THEAD、TFOOT或TBODY所定义的行组中。TR标记符包含<TH>和<TD>标记符，<TH>和<TD>标记符中又包含了表格的实际数据
TH> </TH>	❑ ROWSPAN=Number（单元格所占的行数） ❑ COLSPAN=Number（单元格所占的列数）	TH元素定义了表格中的一个标题单元格，其中的内容通常以黑体显示。TH标记符位于TR标记符内

续表

语　　法	常 用 属 性	说　　明
TH> </TH>	❑ HEADERS=IDREFS（当前单元格的标题单元格列表） ❑ ABBR=Text（标题单元格的缩略形式） ❑ SCOPE [row | col | rowgroup | colgroup]（标题单元格所覆盖的单元格数） ❑ AXIS=CDATA（标题单元格类别） ❑ ALIGN=[left | center | right | justify | char]（单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（单元格的垂直对齐方式） ❑ WIDTH=Pixels（单元格宽） ❑ HEIGHT=Pixels（单元格高） ❑ NOWRAP（单元格内不换行） ❑ BGCOLOR=Color（单元格背景色） ❑ 通用属性	TH元素定义了表格中的一个标题单元格，其中的内容通常以黑体显示。TH标记符位于TR标记符内
<TD> </TD>	❑ ROWSPAN=Number（单元格所占的行数） ❑ COLSPAN=Number（单元格所占的列数） ❑ HEADERS=IDREFS（当前单元格的标题单元格列表） ❑ ABBR=Text（标题单元格的缩略形式） ❑ SCOPE[row | col | rowgroup | colgroup]（标题单元格所覆盖的单元格数） ❑ AXIS=CDATA（标题单元格类别） ❑ ALIGN=[left | center | right | justify | char]（单元格的水平对齐方式） ❑ CHAR=Character（单元格之间的对齐字符） ❑ CHAROFF=Length（对齐字符的偏移量） ❑ VALIGN=[top | middle | bottom | baseline]（单元格的垂直对齐方式） ❑ WIDTH=Pixels（单元格宽） ❑ HEIGHT=Pixels（单元格高） ❑ NOWRAP（单元格内不换行） ❑ BGCOLOR=Color（单元格背景色） ❑ 通用属性	TD元素定义了表格中的一个数据单元格。TD标记符位于TR标记符内

A.1.7　框架元素

语　　法	常 用 属 性	说　　明
<FRAMESET> </FRAMESET>	❑ ROWS=MultiLengths（设置横向框架） ❑ COLS=MultiLengths（设置纵向框架） ❑ ONLOAD=Script（所有框架载入时启动脚本的事件） ❑ ONUNLOAD=Script（所有框架卸载时启动脚本的事件） ❑ 通用属性	FRAMESET元素是一个框架容器，它将窗口分成长方形的子区域，即框架。在一个框架集文档中，<FRAMESET>标记符取代了<BODY>的位置，而紧接<HEAD>标记符之后。FRAMESET标记符中包含一个或多个<FRAMESET>或<FRAME>标记符，并可能含有一个可选的<NOFRAMES>标记符

续表

语　法	常 用 属 性	说　明
<FRAME>	❑ NAME=CDATA（框架名） ❑ SRC=URL（框架的初始页面） ❑ LONGDESC=URL（框架的长篇描述） ❑ FRAMEBORDER=[1 | 0]（设置是否显示框架边框） ❑ MARGINWIDTH=Pixels（边距宽度） ❑ MARGINHEIGHT=Pixels（边距高度） ❑ NORESIZE（禁止修改框架尺寸） ❑ SCROLLING=[yes | no | auto]（设置是否显示滚动条） ❑ 通用属性	FRAME元素定义了一个框架，即一个框架集文档（FRAMESET）中的长方形空间。FRAME标记符必须包含在FRAMESET标记符中
<NOFRAMES> </NOFRAMES>	❑ 通用属性	NOFRAMES元素中包含了框架不能被显示时的替换内容。NOFRAMES元素通常在Frameset文档中使用，它在浏览器不支持框架或框架被禁用时，提供相应的替换内容。NOFRAMES标记符必须位于FRAMESET标记符之间
<IFRAME> </IFRAME>	❑ SRC=URL（框架内容网页的 URL） ❑ NAME=CDATA（框架名） ❑ LONGDESC=URL（到长篇描述的链接） ❑ WIDTH=Length（框架宽度） ❑ HEIGHT=Length（框架高度） ❑ ALIGN=[top | middle | bottom | left | right]（框架对齐方式） ❑ FRAMEBORDER=[1 | 0]（设置是否显示框架边框） ❑ MARGINWIDTH=Pixels（边距宽） ❑ MARGINHEIGHT=Pixels（边距高） ❑ SCROLLING=[yes | no | auto]（是否显示滚动条） ❑ 通用属性	IFRAME元素定义了一个页内框架，可以在其中显示HTML页面。包含在<IFRAME>和</IFRAME>中的内容只有当浏览器不支持框架时才显示

A.1.8 表单元素

语　法	常 用 属 性	说　明
<FORM> </FORM>	❑ ACTION=URL（处理表单结果的脚本的位置） ❑ METHOD=[get | post]（发送表单的 HTTP 方法） ❑ ENCTYPE=ContentType（表单数据的编码类型） ❑ ACCECT-CHARSET=Charsets（可支持的字符列表） ❑ TARGET=FrameTarget（显示表单内容的框架） ❑ ONSUBMIT=Script（表单发送时启动脚本的事件） ❑ ONRESET=Script（表单重置时启动脚本的事件） ❑ 通用属性	FORM元素定义了一个交互式表单。该元素中包含INPUT、SELECT、TEXTAREA和BUTTON等控件，使用户能通过控件与表单传递信息

续表

语　法	常 用 属 性	说　明
<INPUT>	❑ TYPE=[text | password | checkbox | radio | submit | reset | file | hidden | image | button]（控件类型） ❑ NAME=CDATA（控件的名称） ❑ VALUE=CDATA（控件的值） ❑ CHECKED（设置单选按钮或复选框的初始选中状态） ❑ SIZE=CDATA（文本框的宽度，以字符数为单位） ❑ MAXLENGTH=Number（最大文本输入字符数） ❑ SRC=URL（图像源） ❑ ALT=CDATA（图像的替换文本） ❑ USEMAP=URL（客户端图像映射） ❑ ALIGN=[top | middle | bottom | left | right]（表单元素的对齐方式） ❑ DISABLED（使控件无效以防止输入） ❑ READONLY（设置控件为只读） ❑ ACCEPT=ContentTypes（文件上载的媒体类型） ❑ ACCESSKEY=Character（快捷键） ❑ TABINDEX=Number（在 Tab 键遍历次序中的位置） ❑ ONFOCUS=Script（当元素获得焦点时启动脚本的事件） ❑ ONBLUR=Script（当元素失去焦点时启动脚本的事件） ❑ ONSELECT=Script（当文本框中的部分文本被选中时启动脚本的事件） ❑ ONCHANGE=Script（当控件的值改动时启动脚本的事件） ❑ 通用属性	INPUT元素定义了一个用于用户输入的表单控件，通常位于FORM标记符内
<BUTTON> </BUTTON>	❑ NAME=CDATA（控件的名称） ❑ VALUE=CDATA（控件的值） ❑ TYPE=[submit | reset | button]（按钮类型） ❑ DISABLED（使控件无效） ❑ READONLY（设置控件为只读） ❑ ACCESSKEY=Character（快捷键） ❑ TABINDEX=Number（在 Tab 键遍历次序中的位置） ❑ ONFOCUS=Script（当元素获得焦点时启动脚本的事件） ❑ ONBLUR=Script（当元素失去焦点时启动脚本的事件） ❑ 通用属性	BUTTON元素定义了一个按钮，可以是提交、重置或普通按钮。虽然也可以用INPUT元素创建按钮，但用BUTTON元素创建的按钮通常具有更强的表现力
<SELECT> </SELECT>	❑ NAME=CDATA（控件的名称） ❑ MULTIPLE（控制是否可以选择多个选项） ❑ SIZE=Number（显示出的菜单框行数） ❑ DISABLED（使控件无效） ❑ TABINDEX=Number（在 Tab 键遍历次序中的位置） ❑ ONFOCUS=Script（当元素获得焦点时启动脚本的事件） ❑ ONBLUR=Script（当元素失去焦点时启动脚本的事件） ❑ ONCHANGE=Script（当控件的值改动时启动脚本的事件） ❑ 通用属性	SELECT元素定义了一个选项菜单，其中包含若干个OPTGROUP或OPTION元素来为用户提供选项

续表

语　法	常用属性	说　明
<OPTGROUP> </OPTGROUP>	❑ LABEL=Text（组标签） ❑ DISABLED（禁用选项组） ❑ 通用属性	OPTGROUP元素定义了一个SELECT菜单内的选项组，其中至少包含一个OPTION元素来定义实际的选项。注意，多数浏览器并不支持OPTGROUP元素，因此在使用选项菜单时最好直接用OPTION定义选项
<OPTION> </OPTION>	❑ VALUE=CDATA（选项值） ❑ SELECTED（初始选择值） ❑ DISABLED（禁用选项） ❑ LABEL=Text（选项标签） ❑ 通用属性	OPTION元素定义了SELECT菜单中的菜单选项
<TEXTAREA> </TEXTAREA>	❑ NAME=CDATA（控件的名称） ❑ ROWS=Number（多行文本框的行数） ❑ COLS=Number（多行文本框的列数） ❑ DISABLED（使控件无效） ❑ READONLY（设置控件为只读） ❑ ACCESSKEY=Character（快捷键） ❑ TABINDEX=Number（在 Tab 键遍历次序中的位置） ❑ ONFOCUS=Script（当元素获得焦点时启动脚本的事件） ❑ ONBLUR=Script（当元素失去焦点时启动脚本的事件） ❑ ONSELECT=Script（当文本框中的某些文本被选中时启动脚本的事件） ❑ ONCHANGE=Script（当控件的值改动时启动脚本的事件） ❑ 通用属性	TEXTAREA元素定义了一个多行文本框控件
<ISINDEX>	❑ PROMPT（提示信息） ❑ 通用属性	ISINDEX元素定义了一个单行文本输入框。在HTML 4中，它已被INPUT元素取代
<LABEL> </LABEL>	❑ FOR=IDREF（相关表单控件的 ID） ❑ ACCESSKEY=Character（快捷键） ❑ ONFOCUS=Script（元素获得焦点时启动脚本的事件） ❑ ONBLUR=Script（元素失去焦点时启动脚本的事件） ❑ 通用属性	LABEL元素将一个表单控件和一个标签联系起来

续表

语　　法	常 用 属 性	说　　明
<FIELDSET> </FIELDSET>	❑ 通用属性	FIELDSET元素定义了一个表单控件组。通过将相关联的控件分组，可以把表单分为更小、更易于管理的部分，以避免出现用户无法使用过于繁多的控件的情况（注意，并非所有浏览器都支持FIELDSET元素）。在FIELDSET标记符中应包含作为控件组成员的各表单控件，并需要使用LEGEND标记符创建一个控件组标签
<LEGEND> </LEGEND>	❑ ACCESSKEY=Character（快捷键） ❑ ALIGN=[top | bottom | left | right]（标签文字相对于控件组的对齐方式） ❑ 通用属性	LEGEND元素定义了一个控件组的标签，且必须立即出现在<FIELDSET>标记符之后

A.1.9 其他元素

语　　法	常 用 属 性	说　　明
<A> </A>	❑ HREF=URL（链接的目标文件位置） ❑ NAME=CDATA（已命名的链接目标） ❑ REL=LinkTypes（到链接的关系） ❑ REV=LinkTypes（来自链接的关系） ❑ TYPE=ContentType（链接的内容类型） ❑ TARGET=FrameTarget（显示链接的目标框架） ❑ HREFLANG=LanguageCode（链接目标文件的语言） ❑ CHARSET=Charset（链接的字符编码） ❑ ACCESSKEY=Character（快捷键） ❑ TABINDEX=Number（Tab 键遍历次序中的位置） ❑ SHAPE=[rect | circle | poly | default]（客户端图像映射中映射区域的形状） ❑ COORDS=Coords（客户端图像映射中映射区域的坐标） ❑ ONFOCUS=Script（元素获得焦点时启动脚本的事件） ❑ ONBLUR=Script（元素失去焦点时启动脚本的事件） ❑ 通用属性	A元素定义了一个超链接（使用href属性时）或者一个超链接的目的位置（使用name属性时）。当定义超链接时，位于<A>和</A>之间的内容成为超链接的源，浏览者可以单击超链接源跳转到超链接目标

续表

语　法	常 用 属 性	说　明
<APPLET> </APPLET>	❑ CODE=CDATA（类文件名称或路径） ❑ CODEBASE=URL（类文件的基础 URL） ❑ WIDTH=Length（小程序在网页中所占的宽度） ❑ HEIGHT=Length（小程序在网页中所占的高度） ❑ ARCHIVE=URL-LIST（存档文件所在的位置列表） ❑ OBJECT=CDATA（序列化的小程序） ❑ NAME=CDATA（小程序实例的名称，用于小程序间通信） ❑ ALT=Text（替换文本） ❑ ALIGN=[top | middle | bottom | left | right]（小程序在页面的对齐方式） ❑ HSPACE=Pixels（小程序对象左右的空白距离） ❑ VSPACE=Pixels（小程序对象上下的空白距离） ❑ 通用属性	APPLET元素用来嵌入一个Java小程序（Applet）。在HTML 4.0 中，建议使用OBJECT元素代替APPLET元素。使用APPLET标记符时，可以用PARAM标记符指定运行时参数
<AREA>	❑ SHAPE=[rect | circle | poly | default]（客户端图像映射中映射区域的形状） ❑ COORDS=Coords（客户端图像映射中映射区域的坐标） ❑ HREF=URL（链接的目标文件位置） ❑ TARGET=FrameTarget（显示链接的目标框架） ❑ NOHREF（不包含链接） ❑ ALT=Text（替换文本） ❑ TABINDEX=Number（Tab 键遍历次序中的位置） ❑ ONFOCUS=Script（元素获得焦点时启动脚本的事件） ❑ ONBLUR=Script（元素失去焦点时启动脚本的事件） ❑ 通用属性	AREA元素定义了一个在客户端图像映射中的图形区域。AREA标记符位于MAP标记符内
<BASE>	❑ HREF=URL（默认 URL 基准） ❑ TARGEF=FrameTarget（默认目标框架）	BASE元素定义了文档的默认URL基准和默认目标框架。一个文档中最多有一个BASE标记符，而且如果使用，则必须位于HEAD标记符内
<BDO> </BDO>	❑ DIR=[ltr | rtl]（文本的方向） ❑ LANG=LanguageCode（文本的语言） ❑ 通用属性	BDO元素覆盖了所包含文本的双向算法。BDO元素用于设置多语言文本的显示方向，在一般的网页中并不常用
<CENTER> </CENTER>	❑ 通用属性	CENTER元素定义了一个居中对齐的块。在HTML 4.0中，它属过时的用法，通常用 DIV align=center></DIV>代替

续表

语 法	常 用 属 性	说 明
<IMG>	❑ SRC=URL（图像源的位置） ❑ ALT=Text（替换文本） ❑ LONGDESC=URL（包含长篇描述的文档位置） ❑ WIDTH=Length（图像宽度） ❑ HEIGHT=Length（图像高度） ❑ USEMAP=URL（客户端图像映射的映射说明，对应于 MAP 元素指定的内容） ❑ ISMAP（指示使用服务器端图像映射） ❑ ALIGN=top | middle | bottom | left | right（图像对齐方式） ❑ BORDER=Length（图像边框的宽度） ❑ HSPACE=Pixels（图像左右的空白距离） ❑ VSPACE=Pixels（图像上下的空白距离） ❑ 通用属性	IMG元素定义了一个行内图像
<LINK>	❑ REL=LinkTypes（到链接的关系） ❑ REV=LinkTypes（来自链接的关系） ❑ HREF=URL（链接资源的 URL） ❑ TYPE=ContentType（链接的内容类型） ❑ TARGET=FrameTarget（显示链接的目标框架） ❑ MEDIA=MediaDesc（链接的媒体） ❑ HREFLANG=LanguageCode（链接资源的语言） ❑ CHARSET=Charset（链接资源的字符编码） ❑ 通用属性	LINK元素定义了文档的关联关系。LINK标记符应包含在HEAD标记符内，并且可以有多个
<MAP> </MAP>	❑ NAME=CDATA（图像映射的名称） ❑ 通用属性	MAP元素用于定义图像映射的区域信息。MAP的不可缺省的NAME属性通常用作IMG或OBJECT标记符的USEMAP属性的值。MAP标记符内包含多个AREA标记符，用于定义图像上可单击的区域
<NOSCRIPT> </NOSCRIPT>	❑ 通用属性	NOSCRIPT元素为不执行客户端程序的浏览器提供了替代的显示内容。NOSCRIPT标记符应紧跟在它所提供替换内容的SCRIPT标记符后。只有当浏览器不支持客户端程序时，才显示<NOSCRIPT></NOSCRIPT>中的内容

续表

语　法	常用属性	说　明
<OBJECT> </OBJECT>	❑ DATA=URL（对象数据的位置） ❑ CLASSID=URL（实现位置） ❑ ARCHIVE=CDATA（存档文件） ❑ CODEBASE=URL（CLASSID、DATA、ARCHIVE 的基准 URL） ❑ WIDTH=Length（对象宽度） ❑ HEIGHT=Length（对象高度） ❑ NAME=CDATA（如果对象在表单中提交，则定义其名称） ❑ USEMAP=UAL（定义使用的客户端图像映射） ❑ TYPE=ContentType（对象内容类型） ❑ CODETYPE=ContentType（代码内容类型） ❑ STANDBY=Text（装载时显示的信息） ❑ TABINDEX=NUMBER（在 Tab 键遍历顺序中的位置） ❑ DECLARE（声明一个对象而不启动它） ❑ ALIGN=[top | middle | bottom | left | right]（对象对齐方式） ❑ BORDER=Length（对象边框宽度） ❑ HSPACE=Pixels（对象左右的空白距离） ❑ VSPACE=Pixels（对象上下的空白距离） ❑ 通用属件	OBJECT元素在网页中定义了一个对象。这个对象可以是图像、Java小程序、ActiveX控件、多媒体等各种对象。使用OBJECT定义对象时，还可以用PARAM标记符为对象指定运行时参数。在HTML 4.0中，建议用通用的OBJECT元素取代更为特殊的IMG、APPLET等元素。不过，使用OBJECT代替所有其他对象元素（如IMG、APPLET等）的用法目前还没有得到多数浏览器的支持 为确保浏览器的支持，通常使用嵌套的OBJECT元素包含多个对象，以便当浏览器无法显示外层对象时，依次尝试显示内层对象。如果浏览器无法显示Python小程序，则尝试显示MPEG视频；如果仍然无法显示MPEG视频，则尝试显示GIF图像；如果仍然无法显示GIF图像，则显示文本
<PARAM>	❑ NAME=CDATA（参数名称） ❑ VALUE=CDATA（参数值） ❑ VALUETYPE=[data | ref | object]（值的类型） ❑ TYPE=ContentType（当 valuetype=ref 时，指定值的内容类型） ❑ ID=ID（元素的 ID）	PARAM元素指定了对象在运行时需要的一系列值。在OBJECT或APPLET标记符中可以以任意顺序包含任意数量的PARAM标记符。在使用PARAM指定参数时，对象必须能识别所指定的参数名和值
<SCRIPT> </SCRIPT>	❑ TYPE=ContentType（编程语言的内容类型） ❑ LANGUAGE=CDATA（编程语言名） ❑ SRC=URL（外部程序位置） ❑ CHARSET=Charset（外部程序的字符编码） ❑ DEFER（设置此布尔属性时，表示告知浏览器脚本并不产生任何文档内容（例如，在 JavaScript 中没有"document.write"语句），从而使浏览器可以继续解释 HTML 文件的内容并进行显示）	SCRIPT元素在文档中包含一段客户端脚本程序。客户端脚本程序能使文档更好地对客户端的事件作出反应。例如，一段程序可以在用户发送所填写的表单之前先检查用户填写的内容，并立即通知用户填写错误。SCRIPT标记符可以位于文档中的任何位置，但通常位于HEAD标记符内，以便于维护

续表

语　法	常 用 属 性	说　明
<STYLE> </STYLE>	❑ TYPE=ContentType（样式语言的类型） ❑ MEDIA=MediaDesc（应用样式的媒体） ❑ TITLE=Text（样式表的名字）	STYLE元素用于在文档中嵌入样式表。文档的HEAD标记符中可以包含任意数量的STYLE标记符。对于层叠样式表（CSS），TYPE属性的值是text/css。定义样式表时，样式表项的形式为：Selector{property1: value1;property2: value2;...}，其中，Selector可以是HTML标记、样式类、样式ID等，property是由CSS定义的属性，是属性对应的值

A.2　CSS 中支持的颜色名称

在本书中设置颜色时提到过，可以使用颜色的英文名称来设置颜色效果。本附录给出在 CSS 中可以使用的颜色名称、颜色代码以及颜色的效果。

A.2.1　W3C 规定的十六色

W3C 就是 World Wide Web Consortium，全球万维网联盟的简称。它研究 Web 规范和指导方针，致力于推动 Web 发展，保证各种 Web 技术能很好地协同工作。W3C 规定的十六色如表 A.1 所示。

表A.1　W3C规定的十六色

颜色的英文名称	中 文 含 义	十六进制颜色码	十进制RGB颜色值	颜 色 效 果
black	黑	#000000	0,0,0	
white	白	# FFFFFF	255,255,255	
red	红	#FF0000	255,0,0	
yellow	黄	#FFFF00	255,255,0	
lime	酸橙色	#00FF00	0,255,0	
aqua	浅绿色	#00FFFF	0,255,255	
blue	蓝	#0000FF	0,0,255	
fuchsia	紫红色	#FF00FF	255,0,255	
gray	灰色	#808080	128,128,128	
silver	银色	#C0C0C0	192,192,192	
maroon	栗色	#800000	128,0,0	
olive	橄榄色	#808000	128,128,0	
green	绿色	#008000	0,128,0	
teal	水鸭色	#008080	0,128,128	
navy	海军蓝	#000080	0,0,128	
purple	紫色	#800080	128,0,128	

A.2.2 网络安全色

Web 颜色只有 216 色，所以网络上安全色就是指 Web 所能反映、所能支持的 216 色。这里给出表 A.2 是为了让读者更清楚网络中可以正常显示的颜色，从而在创建网页时选择更加合适的色彩。

网络安全色的十六进制颜色码都是可以被 33 整除的值，如#330066、#CC0099 等。

表A.2 网络安全色

十六进制颜色码	十进制RGB颜色值	颜 色 效 果	十六进制颜色码	十进制RGB颜色值	颜 色 效 果
#000000	0,0,0		#990000	153,0,0	
#000033	0,0,51		#990033	153,0,51	
#000066	0,0,102		#990066	153,0,102	
#000099	0,0,153		#990099	153,0,153	
#0000CC	0,0,204		#9900CC	153,0,204	
#0000FF	0,0,255		#9900FF	153,0,255	
#003300	0,51,0		#993300	153,51,0	
#003333	0,51,51		#993333	153,51,51	
#003366	0,51,102		#993366	153,51,102	
#003399	0,51,153		#993399	153,51,153	
#0033CC	0,51,204		#9933CC	153,51,204	
#0033FF	0,51,255		#9933FF	153,51,255	
#006600	0,102,0		#996600	153,102,0	
#006633	0,102,51		#996633	153,102,51	
#006666	0,102,102		#996666	153,102,102	
#006699	0,102,153		#996699	153,102,153	
#0066CC	0,102,204		#9966CC	153,102,204	
#0066FF	0,102,255		#9966FF	153,102,255	
#009900	0,153,0		#999900	153,153,0	
#009933	0,153,51		#999933	153,153,51	
#009966	0,153,102		#999966	153,153,102	
#009999	0,153,153		#999999	153,153,153	
#0099CC	0,153,204		#9999CC	153,153,204	
#0099FF	0,153,255		#9999FF	153,153,255	
#00CC00	0,204,0		#99CC00	153,204,0	
#00CC33	0,204,51		#99CC33	153,204,51	
#00CC66	0,204,102		#99CC66	153,204,102	
#00CC99	0,204,153		#99CC99	153,204,153	
#00CCCC	0,204,204		#99CCCC	153,204,204	
#00CCFF	0,204,255		#99CCFF	153,204,255	
#00FF00	0,255,0		#99FF00	153,255,0	
#00FF33	0,255,51		#99FF33	153,255,51	

续表

十六进制颜色码	十进制RGB颜色值	颜 色 效 果	十六进制颜色码	十进制RGB颜色值	颜 色 效 果
#00FF66	0,255,102		#99FF66	153,255,102	
#00FF99	0,255,153		#99FF99	153,255,153	
#00FFCC	0,255,204		#99FFCC	153,255,204	
#00FFFF	0,255,255		#99FFFF	153,255,255	
#330000	51,0,0		#CC0000	204,0,0	
#330033	51,0,51		#CC0033	204,0,51	
#330066	51,0,102		#CC0066	204,0,102	
#330099	51,0,153		#CC0099	204,0,153	
#3300CC	51,0,204		#CC00CC	204,0,204	
#3300FF	51,0,255		#CC00FF	204,0,255	
#333300	51,51,0		#CC3300	204,51,0	
#333333	51,51,51		#CC3333	204,51,51	
#333366	51,51,102		#CC3366	204,51,102	
#333399	51,51,153		#CC3399	204,51,153	
#3333CC	51,51,204		#CC33CC	204,51,204	
#3333FF	51,51,255		#CC33FF	204,51,255	
#336600	51,102,0		#CC6600	204,102,0	
#336633	51,102,51		#CC6633	204,102,51	
#336666	51,102,102		#CC6666	204,102,102	
#336699	51,102,153		#CC6699	204,102,153	
#3366CC	51,102,204		#CC66CC	204,102,204	
#3366FF	51,102,255		#CC66FF	204,102,255	
#339900	51,153,0		#CC9900	204,153,0	
#339933	51,153,51		#CC9933	204,153,51	
#339966	51,153,102		#CC9966	204,153,102	
#339999	51,153,153		#CC9999	204,153,153	
#3399CC	51,153,204		#CC99CC	204,153,204	
#3399FF	51,153,255		#CC99FF	204,153,255	
#33CC00	51,204,0		#CCCC00	204,204,0	
#33CC33	51,204,51		#CCCC33	204,204,51	
#33CC66	51,204,102		#CCCC66	204,204,102	
#33CC99	51,204,153		#CCCC99	204,204,153	
#33CCCC	51,204,204		#CCCCCC	204,204,204	
#33CCFF	51,204,255		#CCCCFF	204,204,255	
#33FF00	51,255,0		#CCFF00	204,255,0	
#33FF33	51,255,51		#CCFF33	204,255,51	
#33FF66	51,255,102		#CCFF66	204,255,102	
#33FF99	51,255,153		#CCFF99	204,255,153	

续表

十六进制颜色码	十进制RGB颜色值	颜色效果	十六进制颜色码	十进制RGB颜色值	颜色效果
#33FFCC	51,255,204		#CCFFCC	204,255,204	
#33FFFF	51,255,255		#CCFFFF	204,255,255	
#660000	102,0,0		#FF0000	255,0,0	
#660033	102,0,51		#FF0033	255,0,51	
#660066	102,0,102		#FF0066	255,0,102	
#660099	102,0,153		#FF0099	255,0,153	
#6600CC	102,0,204		#FF00CC	255,0,204	
#6600FF	102,0,255		#FF00FF	255,0,255	
#663300	102,51,0		#FF3300	255,51,0	
#663333	102,51,51		#FF3333	255,51,51	
#663366	102,51,102		#FF3366	255,51,102	
#663399	102,51,153		#FF3399	255,51,153	
#6633CC	102,51,204		#FF33CC	255,51,204	
#6633FF	102,51,255		#FF33FF	255,51,255	
#666600	102,102,0		#FF6600	255,102,0	
#666633	102,102,51		#FF6633	255,102,51	
#666666	102,102,102		#FF6666	255,102,102	
#666699	102,102,153		#FF6699	255,102,153	
#6666CC	102,102,204		#FF66CC	255,102,204	
#6666FF	102,102,255		#FF66FF	255,102,255	
#669900	102,153,0		#FF9900	255,153,0	
#669933	102,153,51		#FF9933	255,153,51	
#669966	102,153,102		#FF9966	255,153,102	
#669999	102,153,153		#FF9999	255,153,153	
#6699CC	102,153,204		#FF99CC	255,153,204	
#6699FF	102,153,255		#FF99FF	255,153,255	
#66CC00	102,204,0		#FFCC00	255,204,0	
#66CC33	102,204,51		#FFCC33	255,204,51	
#66CC66	102,204,102		#FFCC66	255,204,102	
#66CC99	102,204,153		#FFCC99	255,204,153	
#66CCCC	102,204,204		#FFCCCC	255,204,204	
#66CCFF	102,204,255		#FFCCFF	255,204,255	
#66FF00	102,255,0		#FFFF00	255,255,0	
#66FF33	102,255,51		#FFFF33	255,255,51	
#66FF66	102,255,102		#FFFF66	255,255,102	
#66FF99	102,255,153		#FFFF99	255,255,153	
#66FFCC	102,255,204		#FFFFCC	255,255,204	
#66FFFF	102,255,255		#FFFFFF	255,255,255	

A.2.3　IE 4 以上版本中的预命名颜色

在 IE 4 以上的版本的浏览器中，有一些颜色可以直接采用英文名称来设置，表 A.3 给出的颜色就是到目前为止，可以直接通过颜色英文名称设置，并能在 IE 4 以上版本的浏览器中正常显示的颜色。

表A.3　在IE 4 以上版本中预命名的颜色

颜色的英文名称	中 文 含 义	十六进制颜色码	十进制RGB颜色值	颜 色 效 果
aliceblue	艾利斯蓝	#F0F8FF	240,248,255	
antiquewhite	古董白	#FAEBD7	250,235,215	
aqua	蓝绿色、浅绿色	#00FFFF	0,255,255	
aquamarine	碧绿色	#7FFFD4	127,255,212	
azure	天蓝色	#F0FFFF	240,255,255	
beige	米色	#F5F5DC	245,245,220	
bisque	桔黄色	#FFE4C4	255,228,196	
black	黑色	#000000	0,0,0	
blanchedalmond	白杏色	#FFEBCD	255,235,205	
blue	蓝色	#0000FF	0,0,255	
blueviolet	蓝色紫罗兰	#8A2BE2	138,43,226	
brown	褐色	#A52A2A	165,42,42	
burlywood	实木色	#DEB887	222,184,135	
cadetblue	军蓝色	#5F9EA0	95,158,160	
chartreuse	黄绿色	#7FFF00	127,255,0	
chocolate	巧克力色	#D2691E	210,105,30	
coral	珊瑚色	#FF7F50	255,127,80	
cornflowerblue	菊蓝色	#6495ED	100,149,237	
cornsilk	米稠色	#FFF8DC	255,248,220	
crimson	深红色	#DC143C	220,20,60	
cyan	蓝绿色、青色	# 00FFFF	0,255,255	
darkblue	深蓝色	#00008B	0,0,139	
darkcyan	深青色	#008B8B	0,139,139	
darkgoldenrod	暗金色	#B8860B	184,134,11	
darkgray	深灰色	#A9A9A9	169,169,169	
darkgreen	深绿色	#006400	0,100,0	
darkkhaki	深褐色	#BDB76B	189,183,107	
darkmagenta	暗红紫色	#8B008B	139,0,139	
darkolivegreen	深橄榄绿色	#556B2F	85,107,47	
darkorange	暗桔黄色	#FF8C00	255,140,0	
darkorchid	深紫色	#9932CC	153,50,204	
darkred	暗红色	#8B0000	139,0,0	
darksalmon	暗肉色	#E9967A	133,150,122	

续表

颜色的英文名称	中 文 含 义	十六进制颜色码	十进制RGB颜色值	颜 色 效 果
darkseagreen	深灰绿色	#8FBC8B	143,188,139	
darkslateblue	深海蓝色	#483D8B	72,61,139	
darkslategray	暗瓦灰色	#2F4F4F	47,79,79	
darkturquoise	深宝石蓝	#00CED1	0,206,209	
darkviolet	暗紫罗兰色	#9400D3	148,0,211	
deeppink	深粉红色	#FF1493	255,20,147	
deepskyblue	暗天蓝色	#00BFFF	0,191,255	
dimgray	暗灰色	#696969	105,105,105	
dodgerblue	闪灰色	#1E90FF	30,144,255	
firebrick	火砖色	#B22222	178,34,34	
floralwhite	花白	#FFFAF0	255,250,240	
forestgreen	森林绿	#228B22	34,139,34	
fuchsia	紫红色	#FF00FF	255,0,255	
gainsboro	淡灰色	#DCDCDC	220,220,220	
ghostwhite	幽灵白	#F8F8FF	248,248,255	
gold	金色	#FFD700	255,215,0	
goldenrod	金麒麟色	#DAA520	218,165,32	
gray	灰色	#808080	128,128,128	
green	绿色	#008000	0,128,0	
greenyellow	黄绿色	#ADFF2F	173,255,47	
honeydew	蜜白色	#F0FFF0	240,255,240	
hotpink	热粉红色	#FF69B4	255,105,180	
indianred	印第安红	#CD5C5C	205,92,92	
indigo	靛青色	#4B0082	75,0,130	
ivory	象牙色	#FFFFF0	255,255,240	
khaki	黄褐色	#F0E68C	240,230,140	
lavender	薰衣草色、淡紫色	#E6E6FA	230,230,250	
lavenderblush	淡紫红色	#FFF0F5	255,240,245	
lawngreen	草绿色	#7CFC00	124,252,0	
lemonchiffon	柠檬稠色	#FFFACD	255,250,205	
lightblue	亮蓝色	#ADD8E6	173,216,230	
lightcoral	亮珊瑚色	#F08080	240,128,128	
lightcyan	亮灰色	#E0FFFF	224,255,255	
lightgoldenrodyellow	亮金黄色	#FAFAD2	250,250,210	
lightgreen	亮绿色	#90EE90	144,238,144	
lightgrey	亮灰色	#D3D3D3	211,211,211	
lightpink	亮粉色	#FFB6C1	255,182,193	
lightsalmon	亮柠檬色	#FFA07A	255,160,122	
lightseagreen	亮水绿色	#20B2AA	32,178,170	

续表

颜色的英文名称	中 文 含 义	十六进制颜色码	十进制RGB颜色值	颜 色 效 果
lightskyblue	亮天蓝色	#87CEFA	135,206,250	
lightslategray	亮瓦灰色	#778899	119,136,153	
lightsteelblue	亮金属色	#B0C4DE	176,196,222	
lightyellow	亮黄色	#FFFFE0	255,255,224	
lime	酸橙色	#00FF00	0,255,0	
limegreen	橙绿色	#32CD32	50,205,50	
linen	亚麻色	#FAF0E6	250,240,230	
magenta	洋红色	#FF00FF	255,0,255	
maroon	栗色	#800000	128,0,0	
mediumaquamarine	中绿玉色	#66CDAA	102,205,170	
mediumblue	中蓝色	#0000CD	0,0,205	
mediumorchid	中粉紫色	#BA55D3	186,85,211	
mediumpurple	中紫色	#9370DB	147,112,219	
mediumseagreen	中灰绿色	#3CB371	60,179,113	
mediumslateblue	中暗蓝色	#7B68EE	123,104,238	
mediumspringgreen	春绿色	#00FA9A	0,250,154	
mediumturquoise	中绿宝石色	#48D1CC	72,209,204	
mediumvioletred	中紫罗兰色	#C71585	199,21,133	
midnightblue	中灰蓝色	#191970	25,25,112	
mintcream	薄荷色	#F5FFFA	245,255,250	
mistyrose	浅玫瑰色	#FFE4E1	255,228,225	
moccasin	鹿皮色	#FFE4B5	255,228,181	
navajowhite	纳瓦白	#FFDEAD	255,222,173	
navy	海军蓝	#000080	0,0,128	
oldlace	老旧缎色	#FDF5E6	253,245,230	
olive	橄榄色	#808000	128,128,0	
olivedrab	深绿褐色	#6B8E23	107,142,35	
orange	橙色	#FFA500	255,165,0	
orangered	红橙色	#FF4500	255,69,0	
orchid	淡紫色	#DA70D6	218,112,214	
palegoldenrod	苍麒麟色	#EEE8AA	238,232,170	
palegreen	苍绿色	#98FB98	152,251,152	
paleturquoise	苍宝石绿	#AFEEEE	175,238,238	
palevioletred	苍紫罗兰色	#DB7093	219,112,147	
papayawhip	番木色	#FFEFD5	255,239,213	
peachpuff	桃色	#FFDAB9	255,218,185	
peru	秘鲁色	#CD853F	205,133,65	
pink	粉红色	#FFC0CB	255,192,203	
plum	洋李色	#DDA0DD	221,160,221	

续表

颜色的英文名称	中 文 含 义	十六进制颜色码	十进制RGB颜色值	颜 色 效 果
powderblue	粉蓝色	#B0E0E6	176,224,230	
purple	紫色	#800080	128,0,128	
red	红色	#FF0000	255,0,0	
rosybrown	褐玫瑰红	#BC8F8F	188,143,143	
royalblue	皇家蓝	#4169E1	65,105,225	
saddlebrown	重褐色	#8B4513	139,69,19	
salmon	鲜肉色	#FA8072	250,128,114	
sandybrown	沙褐色	#F4A460	244,164,96	
seagreen	海绿色	#2E8B57	46,139,87	
seashell	海贝色	#FFF5EE	255,245,238	
sienna	赭色	#A0522D	160,82,45	
silver	银色	#C0C0C0	192,192,192	
skyblue	天蓝色	#87CEEB	135,206,235	
slateblue	石蓝色	#6A5ACD	106,90,205	
slategray	灰石色	#708090	112,128,144	
snow	雪白色	#FFFAFA	255,250,250	
springgreen	春绿色	#00FF7F	0,255,127	
steelblue	钢兰色	#4682B4	70,130,180	
tan	茶色	#D2B48C	210,180,140	
teal	水鸭色	#008080	0,128,128	
thistle	蓟花色	#D8BFD8	216,191,216	
tomato	西红柿色	#FF6347	255,99,71	
turquoise	青绿色	#40E0D0	64,224,208	
violet	紫罗兰色	#EE82EE	238,130,238	
wheat	浅黄色	#F5DEB3	245,222,179	
white	白色	#FFFFFF	255,255,255	
whitesmoke	烟白色	#F5F5F5	245,245,245,	
yellow	黄色	#FFFF00	255,255,0	
yellowgreen	黄绿色	#9ACD32	154,205,50	